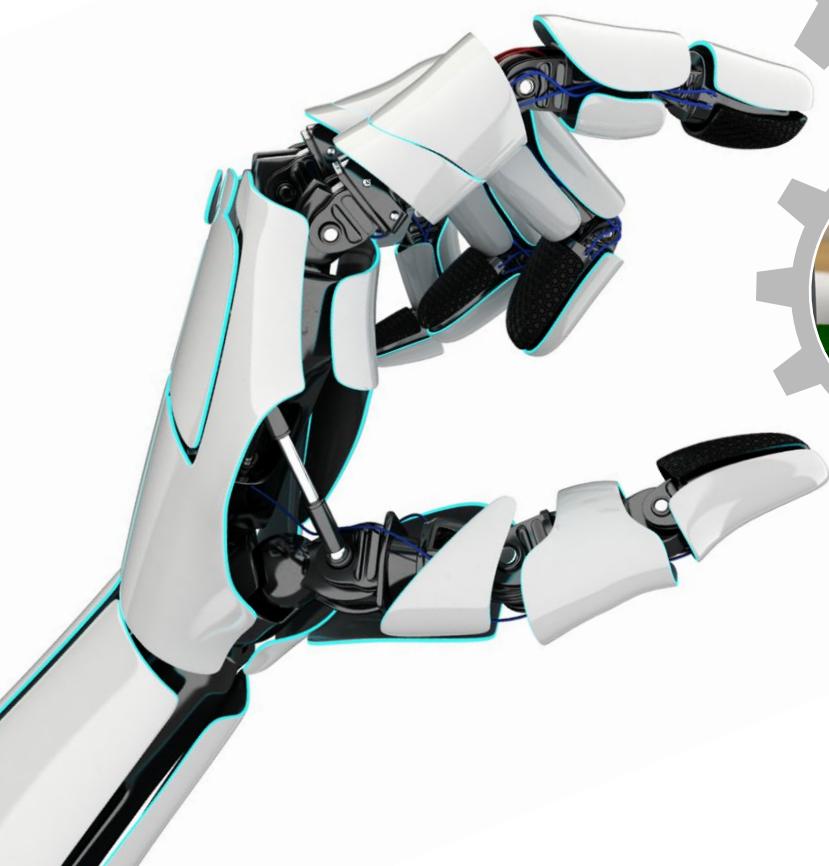
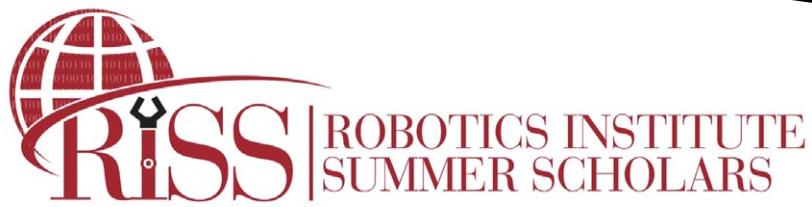


Carnegie Mellon University
The Robotics Institute

Carnegie Mellon University
School of Computer Science

Working Papers Journal

Volume 9 | Fall 2021





Carnegie Mellon Robotics Institute Summer Scholars

Working Papers **Journal**

RISS Director

Dr. John M. Dolan

jdolan@andrew.cmu.edu

RISS Co-director

Ms. Rachel Burcin

rachel@cmu.edu

2021 RISS Journal Team

Jasmine Jerry Aloor

Rachel Moan

Arnab Dey

Elias Rotondo

Iqui Balam Heredia Marin

Manav Singhal

Nikhil Varma Keetha

Nayana Suvarna

Manav Kulshrestha

Renos Zabounidis



We gratefully acknowledge the support of the National Science Foundation through the Research Experience for Undergraduates (REU) program (Grant #1950811).

The Robotics Institute Summer Scholars Working Papers Journal is an annual publication of the Robotics Institute's Summer Scholars Program at Carnegie Mellon University.

Copyright © Carnegie Mellon University 2021.



Thank you to the incredible community who generously gave their time & expertise to guide and mentor the 2021 RI Summer Scholars.

Meet the 2021 RISS Community

Mentors

Abhinav Gupta	Isaac Isukapati	Nicholas Paiva	Stephen Smith
Aishwarya Jadhav	Jack Mostow	Nicolas Mateo Zevallos-Roberts	Sudharshan Suresh
Artur Dubrawski	Jacky Liang	Oliver Kroemer	Teresa Kent
Azarakhsh Keipour	James K. Miller	Peter Schaldenbrand	Tom Bu
Brady Moon	Jay Patrikar	Pragna Mannam	Victoria Dean
Canbo Ye	Jaya Aadityaa	Qin Lin	Vidhi Jain
Carmel Majidi	Ganapathy Subramanian	Reid Simmons	Vikash Kumar
Chen Wang	Jean Oh	Robert Tamburo	Wenhao Luo
Christoph Mertz	Jieshi Chen	Roshni Kaushik	Wenshan Wang
Cornelia Bauer	John M. Dolan	Samuel Triest	Xinjie Yao
Dana Hughes	Joseph Campbell	Sarah Bergbreiter	Yaoyu Hu
Daqian Cheng	Katia Sycara	Saumya Saxena	Yiwei Lyu
Deva Ramanan	Keene Chin	Sayan Mondal	Yu-Xiong Wang
Eliana Cohen	Keith Dufendach	Sebastian Caldas	Yue Guo
Evan Harber	Lu Li	Sebastian Scherer	Yuheng Qiu
Howie Choset	Matthew Travers	Senthil Purushwalkam	Yunzhu Li
Huai Yu	Melisa Orta Martinez	Shu Kong	Zachary Manchester
Huao Li	Michael Kaess	Simone Parisi	Zeynep Temel
Ingrid Navarro	Mononito Goswami	Sophie Yue Guo	Zhongqiang Ren
Inioluwa Oguntola	Nancy S. Pollard	Srinivasa Narasimhan	

Workshop Partners & Presenters

Alexander Hall	Ioannis Gkioulekas	Melisa Orta Martinez	Tekin Meriçli
Andrew "Drew" Bagnell	Jamie Rossi	Neha Shah	Victoria Dean
Bob Frederking	Jean Oh	Oliver Kroemer	Wenzhen Yuan
Changliu Liu	Jennie Piotrkowski	Raunaq Mahesh Bhirangi	Xindi Wu
David Garlan	Jessica Benner	Sarah Bergbreiter	Zachary Manchester
Devdatta Narote	Keely Austin	Shawn Blanton	Zeynep Temel
Erica Weng	Maxim Likhachev	Susan Finger	

Partner Offices

Center for Student Academic Success
School of Computer Science Office of the Dean
Traffic21

University Library Systems
Robotics Institute Business, Administrative, & Creative Offices
CMU Business & Administrative Offices

Special thanks to our design partner, SJH Design!

Table of Contents

A Letter from the Scholars	7
A Letter from Dr. John M. Dolan and Rachel Burcin.	9
Sponsors & Partners.	10
Working Papers	
Transfer Exploration in RL: A Study on Recent Count-Based Methods.	11
<i>Jacob Adkins, Victoria Dean, Simone Parisi, Abhinav Gupta</i>	
Multi-Modal Socially-Aware Imitation Learning for General Aviation	16
<i>Jasmine Jerry Aloor, Jay Patrikar, Sebastian Scherer</i>	
Class-Imbalanced Learning via Bilevel-Optimized Weight Decay.	23
<i>Shaden N Alshammari, Yu-Xiong Wang, Deva Ramanan, Shu Kong</i>	
An Affordable and Accessible Educational Manipulator.	30
<i>Samuel C. Alvares, Pragna Mannam, Oliver Kroemer, Zeynep Temel</i>	
Synthetic Data Generation for the Natural Language Component of an Artificial Social Intelligence Agent.	37
<i>Ryan Aponte, Aishwarya Jadhav, Joseph Campbell, Dana Hughes, Katia Sycara</i>	
Trajectory Planning for a UAV Wrench Task Considering Vehicle Dynamics and Force Output Capabilities.	42
<i>Andrew Ashley, Azarakhsh Keipour, Sebastian Scherer</i>	
User Centered Approach for Developing a Robot Assisted Femoral Vascular Access Device for the Battlefield	46
<i>Lama Bahanan, Nicolas Mateo Zevallos-Roberts, Howie Choset</i>	
Multi-agent Coordination for Automation of U.S. Air Force Installed Systems Testing.	49
<i>Lauren J Blanks, Isaac K Isukapati</i>	
Growing Compliant Mechanisms from Mycelial Materials	55
<i>Danielle Brennan, Keene Chin, Carmel Majidi</i>	
Automatic Detection of Road Work and Construction with Deep Learning Model and a Novel Dataset.	59
<i>Brian Chen, Robert Tamburo, Srinivasa Narasimhan</i>	
Joint SLAM on Multiple Monocular Cameras for Legged Robots	66
<i>Thomas Detlefsen, Sayan Mondal, Matthew Travers</i>	
Weakly Supervised Classification of Vital Sign Alerts as Real or Artifact	70
<i>Arnab Dey, Mononito Goswami, Joo Heung Yoon, Gilles Clermont, Michael Pinsky, Marilyn Hravnak, Artur Dubrawski</i>	
Tree Modelling for Robotic Manipulation using a 3D Autoencoder	78
<i>Christian Eberle, Oliver Kroemer</i>	
Sparse Reconstruction for Unsupervised Video Object Segmentation	83
<i>Daniel Ekpo, Senthil Purushwalkam, Abhinav Gupta</i>	
Probabilistic Safe Reinforcement Learning using Control Barrier Function for Autonomous Vehicle Ramp Merging Control	89
<i>Quanzhi Fu, Yiwei Lyu, John M. Dolan</i>	
Automatic Multi-modal Calibration of Stereo Cameras, Thermal Cameras, and Lasers in Arbitrary Scenes	96
<i>Taimeng Fu, Huai Yu, Yaoyu Hu, Sebastian Scherer</i>	
Reason & Act : A Modular Approach to Explanation Driven Agents for Vision and Language Navigation	102
<i>Shaunak Halbe, Ingrid Navarro, Jean Oh</i>	
Accurate Pedestrian Localization for Urban Crosswalks.	108
<i>Rayna Hata, Issac Isukapati, Stephen Smith</i>	
GPU Enhanced Front-end for Visual-inertial Odometry	112
<i>Yao He, Huai Yu, Sebastian Scherer</i>	

Table of Contents

Continued

AlgeGloves: An Interactive Algebra Interface that Allows Students to Mold Algebraic Functions	118
<i>Iqui Balam Heredia Marin, Melisa Orta Martinez</i>	
Delay-aware Control for Safe Autonomous Driving	123
<i>Dvij Kalaria, Qin Lin, John M. Dolan</i>	
AirObject: An Evolving Topological Graph-based Object Encoding for Semantic Loop Closure	130
<i>Nikhil Varma Keetha, Chen Wang, Yuheng Qiu, Sebastian Scherer</i>	
Solving CommonRoad Motion Planning Benchmarks using CILQR	137
<i>Shivesh Khaitan, Qin Lin, John M. Dolan</i>	
Modelling Human Trust in Commanding of Robot Wingmen	144
<i>Manav Kulshrestha, Huao Li, Konstantinos Mitsopoulos, Dana Hughes, Katia Sycara</i>	
Tuning Mechanical and Aerodynamic Properties of Micro Whisker Sensors for Improved Airflow Detection and Stimuli Sensing	148
<i>Courage Lahban, Teresa Kent, Sarah Bergbreiter</i>	
Replicating Bugs Faster	155
<i>Evolone Layne, Jack Mostow</i>	
Multi-agent Hierarchical Reinforcement Learning in Urban and Search Rescue	157
<i>Long Le, Dana Hughes, Katia Sycara</i>	
SiamFIND: Towards One-shot Informed InterestiNg Object Detection	165
<i>Bowen Li, Chen Wang, Sebastian Scherer</i>	
Self-learning of Crawling Behavior for a Modular Quadruped Robot with Bayesian Optimization	171
<i>Chuan Li, Lu Li, Howie Choset</i>	
Understanding and Predicting Human Activities in Search and Rescue Tasks	174
<i>Ruiyu Li, Yue Guo, Dana Hughes, Katia Sycara</i>	
Generating Stenosis Regions of Interest in X-Ray Coronary Angiography using Deep Neural Networks	179
<i>Jennifer Liu, James K. Miller, Keith Dufendach, Artur Dubrawski</i>	
Unsupervised Visual to Thermal Image Translation and Registration	184
<i>Xinhang Liu, Huai Yu, Sebastian Scherer</i>	
Gander: A Comprehensive Machine Learning Synthesized Media Evaluation Platform & Style-CLIPDraw: A Style-Infused Text-to-Drawing Synthesis Method	189
<i>Zhixuan Liu, Peter Schaldenbrand, Jean Oh</i>	
Development of a Force Sensing Glove for the Analysis of Dynamic Motions	196
<i>Molly Loughney, Cornelia Bauer, Nancy S. Pollard</i>	
Randomized Approach to Informative Path Planning for Multiple UAVs	202
<i>Rachel Moan, Brady Moon, Sebastian Scherer</i>	
Facial Expressions in Quori Using Gazebo and ROS	206
<i>Allison Moore, Roshni Kaushik, Reid Simmons</i>	
Learning a CBF controller for Adaptive Cruise Control Under Model Uncertainty	212
<i>Emanuel Munoz, Qin Lin, John M. Dolan</i>	
Informative and Fast Exploration Planning Using UAV for Reconnaissance Operations	219
<i>Kaleb Ben Naveed, Brady Moon, Sebastian Scherer</i>	
Human Detection, Classification & Tracking in Context of Transit Systems	226
<i>Chigozie Ofodike, Christoph Mertz</i>	
Control System Modeling for Closed-Loop Controlled Ventilator using MATLAB Simulink	231
<i>Zaria Oliver, Lu Li, Howie Choset</i>	

Table of Contents

Continued

Model-Based Reinforcement Learning for Off-Road Navigation	235
<i>Ashley Peake, Samuel Triest, Wenshan Wang, Sebastian Scherer</i>	
ROCCER Evaluation Within Federated Classifier Selection for Improved Identification of Center Collaboration Opportunities	240
<i>Willa Potosnak, Sebastian Caldas, James K. Miller, Artur Dubrawski</i>	
Learning with Noisy Camera Extrinsics for Robust and Real-Time Omnidirectional Depth Prediction.	247
<i>Conner Pulling, Yaoyu Hu, Sebastian Scherer</i>	
Detecting and Classifying Waste Bin Garbage Levels Along Transit Bus Routes	253
<i>Elias Rotondo, Christoph Mertz</i>	
Generalization in Reinforcement Learning through Representation.	259
<i>Rutav Shah, Vikash Kumar, Yunzhu Li, Abhinav Gupta</i>	
Discovering Affordances from Visual Perception in the Carnegie Mellon Ballbot	265
<i>Dimitria Silveria, Oliver Kroemer, Saumya Saxena</i>	
Explanations in Multi-Agent Search and Rescue Task	270
<i>Manav Singhal, Vidhi Jain, Dana Hughes, Katia Sycara</i>	
Evaluation of AutoML Systems Using OpenML Datasets	274
<i>Maya Sitaram, Jieshi Chen, Artur Dubrawski</i>	
Ensembles for Online Dynamics Modeling of Off-road Terrain.	284
<i>Matthew Sivaprakasam, Samuel Triest, Wenshan Wang, Sebastian Scherer</i>	
Multi-robot Simulation Dataset to Analyze Distributed Inference Under Perceptual Aliasing.	289
<i>Thomas Snyder, Sudharshan Suresh, Michael Kaess</i>	
Concept Whitening for Transfer of Concepts in Imitation Learning	295
<i>Grace Su, Ini Oguntola, Katia Sycara, Dana Hughes, Joseph Campbell</i>	
Multi-Agent Path Finding for Convex Polygon-Shaped Agents	303
<i>Nayana Suvarna, Zhongqiang Ren, Howie Choset</i>	
Optimal Control for High-Relative-Degree Systems Under Uncertainty using Exponential Control Barrier Functions.	307
<i>Spencer Van Koevering, Yiwei Lyu, Wenhao Luo, John M. Dolan</i>	
Context-Sensitive Template-Based Text Generation for Robotic Agents	316
<i>Peter Angel Vanegas, Roshni Kaushik, Reid Simmons</i>	
Toward the Use of Model Predictive Control for Quadrotor Flight.	320
<i>Fausto Vega, Zachary Manchester</i>	
Semantic Segmentation in Complex Scenes for Robotics Navigation	325
<i>Zhanxin Wu, Xinjie Yao, Jean Oh</i>	
Concept Whitening for Interpretability in Multi Agent Reinforcement Learning	330
<i>Renos Zabounidis, Joseph Campbell, Ini Oguntola, Dana Hughes, Katia Sycara</i>	
Drone Camera Calibration via Neural and Heuristic Method	334
<i>Longwen Zhang, Yaoyu Hu, Sebastian Scherer</i>	



From the Scholars...

The Robotics Institute Summer Scholars Working Papers Journal is an annual publication of the Robotics Institute's Summer Scholars Program at Carnegie Mellon University.

The journal is a medium for the undergraduate students of the summer research program to communicate their work in collaboration with the participating lab faculties. This journal encompasses the learnings and research findings of the students over the eleven-week-long remote engagement with the CMU community for the year 2021.

The journal comprises 58 papers written by the scholars participating in RISS 2021. The papers included exploring varied domains of Robotics, including Localization, Mapping, Computer Vision, Motion-planning, Controls, Haptics, Aerial Systems, Medical Robotics, Multi-agent Systems, Machine Learning, and Reinforcement Learning. The papers have been drafted by the scholars in collaboration with graduate students and faculty mentors.

The Journal Team would like to acknowledge the splendid efforts put in by the scholars for the papers and for making the peer-review process a success enhancing the quality of the papers.

The scholars would like to thank all the mentors for their invaluable guidance and feedback throughout the program. Their expertise has been of immense support for the scholars.

The scholars would also like to acknowledge the support provided by Dr. Keely Austin, Ms. Jamie Rossi, and the Center for Student Academic Success (CSAS). The CSAS held several workshops and one-on-one appointments for individual scholars throughout the program. Their assistance in guiding and reviewing the individual works has helped the scholars to acquire the necessary skills for writing and presenting their work. We are grateful to Mr. Alexander Hall and the CSAS team for their support. Furthermore, we would like to thank Library Liaison Dr. Jessica Benner for guiding us through the CMU library resources.

Finally, the cohort would like to thank RISS co-directors Ms. Rachel Burcin and Dr. John M Dolan, and RISS summer 2021 partner Ms. Jennie Piotrkowski, who have put their efforts into making this program possible even in the most challenging time of the global pandemic. Their ability to successfully coordinate the program and welcome each scholar to the CMU community while overcoming the barriers introduced by the remote program was inspiring. The RISS experience would not have been possible without their hard work and coordination. We are also thankful to the whole CMU community for their contribution to the program.

— *The RISS Scholar Journal Team*



**58 SCHOLARS
FROM 43 COUNTRIES
& 33 UNIVERSITIES**



Dear Colleagues

Welcome to the 2021 Carnegie Mellon Robotics Institute Summer Scholars (RISS) Working Papers Journal.

The CMU RISS community supports opportunities for students from across the country and around the world to conduct research with robotics researchers at CMU. We are pleased to present this collection of articles that reflect the range of topics and diversity of research to which the scholars contributed.

The RISS undergraduate research program is diverse, global, and inclusive.

Launched in 2006, RISS provides opportunities for students from across the country and around the world to conduct research with CMU leaders in robotics and artificial intelligence. Scholars build knowledge, skills, and a network that will open doors for years to come. We work to create an atmosphere where students can explore and develop their identities as scientists. RISS is a community that cares. It fosters a diverse and inclusive working and learning environment where all students are actively welcomed, included, and supported. The RISS community has hosted students from over 60 home countries plus cities and towns across the United States for research experiences.

The RISS 2021 cohort was the largest ever.

- 58 scholars (29 US Scholars)
- 43 home universities
- 13 countries of citizenship

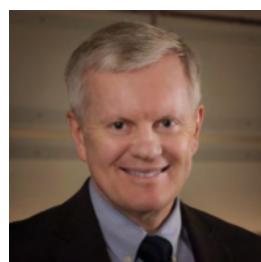
The 2021 cohort was selected from an applicant pool of over 700 applications from more than 40 countries and over 300 institutions worldwide. Of the 29 scholars from the United States, over 60% came from a community underrepresented in STEM.



Sponsors, mentors, and partners make a tremendous difference in the lives of these scholars. Sponsors create access, opportunity, and impact. Additionally, the School of Computer Science and the Robotics Institute provided tremendous support to ensure emerging scholars have this opportunity despite the added challenges of the COVID-19 pandemic. ***More than 100 members of the extended CMU community worked together to shape the 2021 scholar learning experience, including 6 RISS alumni.***

With gratitude,

John & Rachel



Dr. John M. Dolan
Director of RISS Program
& Principal Systems Scientist
jdolan@andrew.cmu.edu



Ms. Rachel Burcin
Co-Director of RISS Program &
Global Programs Manager
rachel@cmu.edu

Thank You

Program Sponsors & Partners



We gratefully acknowledge the support of the National Science Foundation IIS Div of Information & Intelligent Systems through the Research Experience for Undergraduates (REU) program (Grant # 1950811).

Carnegie Mellon University
The Robotics Institute

Carnegie Mellon University
School of Computer Science



برنامـج جامـعة الـملك عـبدالـله
لـلـعلوم والـتقـنيـة لـلـطلـبة الـموهـوبـين
KAUST GIFTED STUDENT PROGRAM



香港中文大學(深圳)
The Chinese University of Hong Kong, Shenzhen



上海科技大学
ShanghaiTech University

AF
UNITED STATES
AIR FORCE ACADEMY

Mobility21
A USDOT NATIONAL
UNIVERSITY TRANSPORTATION CENTER

Carnegie Mellon University
Student Academic Success Center

Carnegie Mellon University
University Libraries

Carnegie Mellon University
Mechanical Engineering

Transfer Exploration in RL: A Study on Recent Count-Based Methods

Jacob Adkins¹, Victoria Dean^{*2}, Simone Parisi^{*3}, Abhinav Gupta^{2,3}
* equal contribution

Abstract—Task agnostic exploration in reinforcement learning is an important research area, especially in settings with sparse or unknown rewards. Learning efficient exploration in these settings remains a key challenge. Existing exploration methods define intrinsic motivation in a multitude of ways, including visitation counts, learned models, and environment changes. We build on a recent paradigm shift in how we learn and evaluate task agnostic exploration: an agent should first learn to explore by training a task-free exploration policy and later transfer this policy to explore unseen environments while learning to solve tasks. This transfer setting provides a clear opportunity for combining multiple forms of exploration across phases of training. We consider the recently developed method of Count-Based Exploration Transfer (C-BET) and extend it to store a record of its state change counts, used as an auxiliary task for the extrinsic policy at transfer time. We present experiments in the procedurally-generated MiniGrid setting with this extended version and compare to the original method. We then extend to the recently released robotics simulator Habitat 2.0, presenting initial experiments using a random policy over a continuous action space and discuss how we plan to utilize this simulator to test C-BET in the robotics setting of a home assistance robot.

Index Terms—Reinforcement Learning, Exploration, Transfer Learning

I. INTRODUCTION

In the context of reinforcement learning, exploration is taxonomized into two broad categories: task-driven and task-agnostic exploration. In task-driven exploration, the agent maximizes the expected discounted sum of future rewards that it receives for some extrinsically motivated task. Whereas, in task-agnostic exploration, the agent learns to explore its environment without any external reward, instead relying on some definition of intrinsic motivation which to some extent may be more similar to how humans learn. When we are babies, we learn to interact with and explore objects in our environment with nothing but our own intrinsically motivated curiosity driving us [1]. This initial exploration is not for a specific goal but still serves us well when we attempt other extrinsically motivated tasks [2]. Some prior definitions of intrinsic reward have included prediction error of a concurrently-learned forward dynamics model [3], [4], action impact evaluated as environment change [5], and states rarity as measured by state visitation counts [6], [7].

¹Jacob Adkins is with the Departments of Mathematics and Computer Science, New College of Florida, Sarasota, FL 34243, USA
jacob.adkins18@ncf.edu

²Dean and Gupta are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA
vdean,gabhinav@andrew.cmu.edu

³Parisi and Gupta are with Facebook AI Research, Pittsburgh, PA 15213, USA
sparisi@fb.com

One recent paradigm shift in task-agnostic exploration moves away from learning and evaluating in a single environment; instead, an agent should first learn a task-free exploration policy across many environments and later transfer this policy to explore unseen environments where it learns to solve tasks.

Within this new transfer framework, we build from previous work that introduces Change-Based Exploration Transfer (C-BET). [8] This method combines state visitation and state change counts for an intrinsic reward. C-BET is based on the idea that some objects are inherently interesting and an agent may wish to interact more with these objects than a method that only considers state visitation counts would allow. The state change count is a way to capture the “interestingness” of an object. C-BET learns an intrinsic exploration policy which is then added as a fixed bias to the task-specific policy to encourage exploration.

This paper documents a proposed extension to C-BET and proposes future work with C-BET in continuous action spaces. Firstly, we note that current intrinsically motivated exploration methods are quite lacking in sample efficiency. Using real robots is also very expensive. Therefore, the amount of data our agent is able to learn from may be limited, and sample efficiency is of the utmost importance. C-BET utilizes transfer learning to improve sample efficiency. However, the two-stage setup of C-BET limits the agent to focusing on the extrinsic task at transfer time. What if the agent not only cares about this task but also might want to do more things in the future? We may wish for the agent to *both* perform well at its task *and* continue intrinsically motivated exploration.

We look at the transfer paradigm proposed in [8], but extend this paradigm further, asking the question: what happens when an agent not only uses a pre-trained exploration policy but continues relying on intrinsic motivation while learning a task as well? This agent will attempt to both solve its extrinsic task and maintain curiosity about the world.

To this end, we propose transferring not only the previously learned policy but also a record of the state change counts from the previous environment(s) with the hope that saving the counts may be useful for the agent and cut down on the number of new samples required. We still fix the exploration policy but use the saved counts as a reward bonus to the extrinsic reward policy.

Secondly, we note that in many real-life scenarios, the action space will be continuous rather than discrete. There are many continuous control domains, such as robotic manipulation or autonomous driving. Continuous action spaces

are difficult since the number of states is infinite and an exhaustive search is not possible. We explore how to extend and evaluate intrinsic motivation methods to continuous action spaces.

II. RELATED WORKS

A. Intrinsic Motivation

One approach to exploration in sparse reward environments is to define a form of intrinsic motivation for the agent to augment the reward signal. There have been several attempts in recent years to define different forms of intrinsic motivation. Some methods have used visitation count bonuses [6], [9]. Others have used prediction error error of some learned model [3], [8], [10], [11]. The C-BET paper [8], whose method we extend and evaluate, notes that these approaches are agent-centric (i.e. based on the agent's belief) and encourages also utilizing environment-centric ideas when forming exploration policies.

B. Transfer

Transfer learning is the idea that agents should be able to use their experience from previous environments to help in their new environment. This could be a learned policy [12]–[15] or new some pre-learned state representation [16], [17]. C-BET proposes approaching transfer in a task-agnostic way by learning a task-free exploration policy that is transferred to the new environment and used as a fixed bias to a policy motivated by learning from an extrinsic reward. [8]

In the first part of this work, we examine whether it is advantageous for the agent to transfer not only this exploration policy but also a record of the change counts to assist as a reward bump in the learning of the second task-specific policy.

III. PRELIMINARIES

We consider the classic reinforcement learning formulation governed by a Markov Decision Process (MDP). At each time step t , the agent evaluates its state s_t , takes an action a_t according to its policy $\pi(a|s)$, the environment transitions to a new state s'_t , and the agent receives a reward signal r_t . The agent's objective is to take actions such that it maximizes its expected sum of rewards received during its episode. The agent learns from its experiences to develop its policy π : a distribution on actions given state, and its value function Q : an estimate of the expected future reward given the state and action.

An MDP can formally be described by a tuple: (S, A, P, R, γ) , where S is the state space, A is the action space, $P(s'|a, s)$ is a distribution that describes the probability of the agent arriving in state s' given that the action a and previous state s , $R(s, a)$ is a distribution on the reward that the agent receives when taking action a in state s and γ is a discount factor for determining the value of future reward. The agent must balance what is known as the exploration-exploitation dilemma. Since environment dynamics are unknown, the agent must choose whether to take explore (i.e. take actions to obtain more information

about its environment) or exploit (i.e. take actions that the agent expects to provide the highest future reward given its current knowledge of the world). The RL problem becomes especially difficult when the rewards are sparse. This occurs when the agent only gets reward at the end of a task, such as winning a game or finding a location in a maze. Often the state and action spaces are far too large to exhaustively search and the agent still must decide how to explore with a very faint (or no) reward signal.

IV. C-BET OVERVIEW

Change-Based Exploration Transfer (C-BET) works in two phases. First, the agent learns an exploration policy in a task-agnostic way from interacting in one or multiple environments. Second, it transfers this learned exploration policy to a new environment to assist in solving the new extrinsic task.

During the pre-training phase, the agent learns an exploration policy $\pi_{exp}(a|s) = \sigma(Q_i(s, a))$ by seeking to maximize its intrinsic reward. This reward is based on two components: an agent-centric count of states and an environment-centric count of changes between states when the agent takes an action. The intrinsic reward is defined as

$$r_{i,t} = 1/\sqrt{N(c(s, a, s'))N(s')}$$

Where $c(s, a, s')$ is the change of a transition (s, a, s') and N is a (pseudo)count of changes and states. When a pseudo count is needed, as in the case of continuous state spaces, C-BET uses #Exploration [18] with SimHash [19] to map images to hash codes and count their occurrences with a hash table.

During the transfer phase, the agent fixes the the interest-value function that it learned during pre-training $Q_i(s, a)$ and learns a new one $Q_e(s, a)$ from only extrinsic rewards. In this phase, the policy is $\pi_{task}(a|s) = \sigma(Q_i(s, a) + Q_e(s, a))$ The idea here is that early on the agent will follow only the interest-value function since extrinsic rewards are sparse but as the agent collects more extrinsic rewards it will become greedier with respect to the extrinsic reward value function.

V. CONTINUING INTRINSIC MOTIVATION AT TRANSFER TIME

Our work considers what happens when, instead of only learning from the extrinsic rewards at transfer, we continue to learn from intrinsic rewards, but with a record of all of the change counts from the previous pre-training environment. The intuition for transferring the change counts specifically is that the change counts represents how the environment changes, so learning to maximize rare changes may make the policy interested in interactive states/actions. If the agent learns that pushing a button or opening a door is interesting, the agent will want to remember it. In contrast, state counts just tell the agent to visit different states. Since during pre-training there is no reward, at transfer the agent may need to revisit states the agent has seen during pretraining. If state counts were stored, the agent would not revisit those states. The interest exploration policy $Q_i(s, a)$ is still fixed and used

as a bias but now our new value function $Q_{i,e}(s, a)$ will be learned from the sum of the intrinsic and extrinsic rewards $r_t = \alpha r_{it} + \beta r_{et}$ where r_{it} is the intrinsic reward at time t and r_{et} is the extrinsic reward at time t with alpha and beta being hyper-parameters. In the intrinsic part of the reward, the state counts are reset at the beginning of transfer to the new environment while the change counts are transferred from the previous environment.

VI. EXPERIMENTS

A. Experimental Setup

For evaluating the proposed extension of transferring the change counts, we experimented in several MiniGrid environments which are procedurally generated gridworlds. The agent can move between discrete squares in the grid and interact with objects such as keys, doors, boxes, etc. This is an interesting setting for exploration because rewards are sparse and often tasks can require many steps before the goal is reached (e.g. move box, pickup key, use key to unlock door, then find gold coin). In MiniGrid, there are many environments of varying difficulties. The experiments that were run to follow up on the MiniGrid experiments run in the C-BET paper [8]. We repeat several of the experiments performed but with the transferred state change counts and learn the after transfer policy as based on the sum of the intrinsic and extrinsic rewards to see how the agent performs. The agent sees a $7 \times 7 \times 3$ partial observation space where it can see the 7×7 tiles in front of it. However, it cannot see behind walls or doors or inside of boxes. The action space available to the agent is discrete with seven actions: left, right, forward, pick up, drop, toggle, and done. The change encoding is a binary vector saying which parts of the state vector have changed after an action $a(s, a, s') := [s_1 \neq s'_1, s_2 \neq s'_2, \dots]$.

First, the agent was pre-trained in three different environments: KeyCorridorS3R3, BlockedUnlockPickup, and MultiRoom-N5-S5. This is where the agent learned its task-free exploration policy and saved the counts of the state change counts. Following this, the agent is transferred to its next environment where we measure its performance by the extrinsic reward it received for achieving the task. The transfer environments used were Unlock, BlockedUnlockPickup, KeyCorridorS3R3, MultiRoom-N6, MultiRoom-N12-S10, and ObstructedMaze-1D1h.

B. Experiment Results

The results were similar across environments so here we present the results of only two of the environments: BlockedUnlockPickup and MultiRoom-N6. In BlockedUnlockPickup, the agent must pick up the ball in front of the door, drop it somewhere else, pick up the key, unlock the door, and then open the box in the other room. In MultiRoom-N6, the agent must navigate through six rooms of maximum size ten and go to the green goal. The plots shown here are of interactions per episodes (a metric of exploration performance) of the agent after transfer as well as the extrinsic reward received.

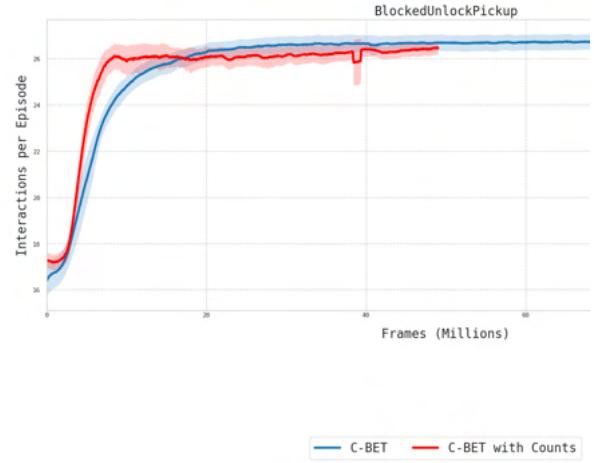


Fig. 1. BlockedUnlockPickup Interactions

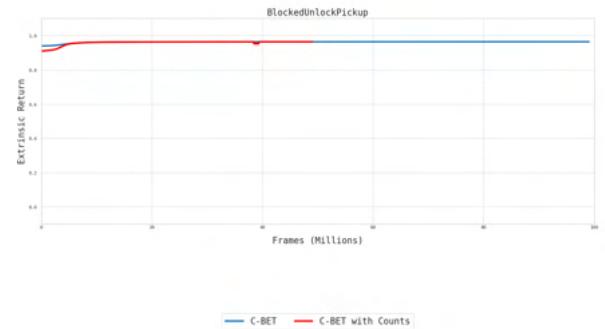


Fig. 2. Blocked Unlock Pickup Extrinsic Return

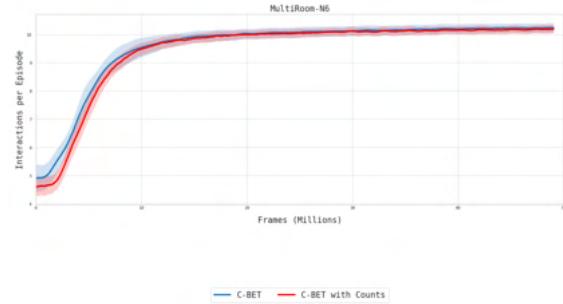


Fig. 3. MultiRoom-N6 Interactions

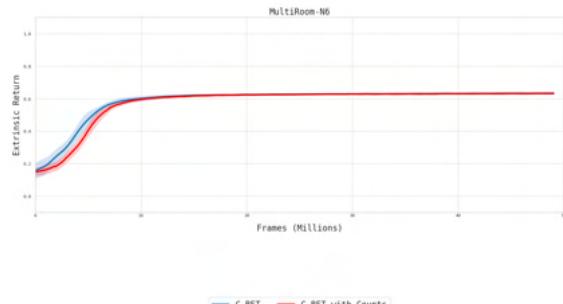


Fig. 4. MultiRoom-N6 Extrinsic Return

The results demonstrate that transferring the state change counts does not improve performance on the extrinsic task nor exploration as measured by interactions per episode.

VII. MOVING INTRINSIC MOTIVATION BEYOND GRIDWORLDS: HABITAT 2.0

A. Initial Continuous Action Experiments

In Habitat 2.0, there are both discrete and continuous state and action spaces. In the continuous setting, there is a rearrange task where the agent is a Fetch robot as seen in Fig. 6. The robot is placed in a room with a goal object and clutter randomly spawned in 6 receptacles. The robot is tasked with picking up the object and returning its arm to the resting position using data from an RGB camera. The agent takes actions in a 7-dimensional joint space and controls the gripper using discrete grasp and release actions.



Fig. 5. A mobile Manipulator (Fetch Robot) in Habitat 2.0 performing rearrangement tasks. Taken from Figure 1. of Habitat 2.0 paper [20]

B. Experimental Setup

For an initial experiment, we visualize the performance of randomly exploring the action space by sampling actions uniformly over the joint space at each step in the episode. We consider two cases: when the robot begins each episode in the same position in front of the object and also when Gaussian noise is added to perturb the starting location of the base position and angle. The Gaussian distributions have a mean 0 and variances of 0.4 and 1.2 for the base position angle respectively. Fig. 6 and Fig. 7 visualize the end-effector position data for 10,000 episodes color mapped by time with no noise added and noise added respectively. Dark blue denotes the beginning of the episodes and red denotes the end.

C. Results

Fig. 8 and Fig. 9 display a density plot of the end-effector states that are visited over 10,000 episodes with no noise added and noise added respectively.

Figures 6-9 show how challenging exploration is in real-world problems. In RL, usually the agent starts by following random actions in the hope to find rewards that will guide it. However, Figure 8 and 9 show how meaningless a random policy would be in a real scenario: the agent ends up visiting

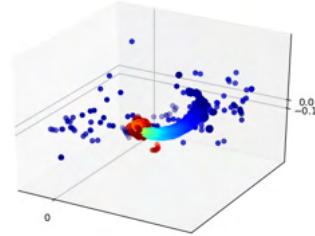


Fig. 6. End-effector position of agent taking random policy with over the joint space for 10,000 episodes colormapped by time (no Gaussian noise)

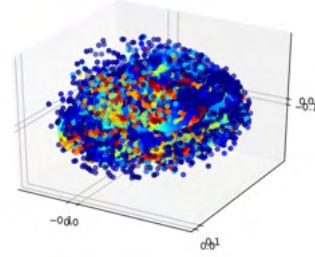


Fig. 7. End-effector position of agent taking random policy with over the joint space for 10,000 episodes colormapped by time (with Gaussian noise)

the same states over and over, possibly failing to even complete full trajectories due to hardware constraints. Fig 6 and 7 show that late-in-time steps always converge to the center of the reachable area, and this is confirmed by the density plots in Fig. 8 and 9. This highlights the need for more intelligent exploration strategies.

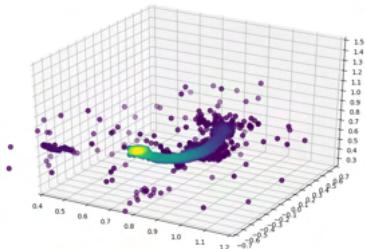


Fig. 8. Density of states visited by random policy (no Gaussian noise)

D. Future Work with Continuous Actions

The work in Habitat 2.0 is still in very early stages. There are several ideas that we have for where to proceed with experiments. First, we would like to see how well C-BET extends to the 7-dimensional continuous action space with the discrete grasping action. In addition, when the code for the end-effector and velocity controls is released, we would like to try C-BET in the 3-dimensional end-effector action

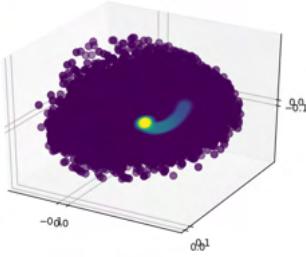


Fig. 9. Density of states visited by random policy (with Gaussian noise)

space with the discrete grasping action. We expect change-based counts to be more effective than state-based counts, as there are many states that involve the robot arm floating in space and not interacting with the world. Finally, when Habitat 2.0 releases Home Assistance Benchmark (HAB) [20], their suite of higher-level tasks and actions, we would like to test C-BET on that and compare it to other benchmarks. For experimentation, we are considering both basic task-agnostic exploration within the same environment as well as pre-training and transfer between scenes. Additional inquiry into transfer of counts and/or value functions could be worth exploring in the new environment.

VIII. CONCLUSIONS

Sample efficient exploration and intrinsic motivation are crucial to successful robot learning in the real world. There are numerous applications in home assistance, medicine, shipping, and countless other fields. We hope to begin investigating how to effectively approach sample efficient exploration in real-world settings.

ACKNOWLEDGMENT

The author (Jacob Adkins) would like to extend sincere gratitude to Victoria Dean, Simone Parisi, and Abhinav Gupta for providing invaluable mentoring and guidance and for directing this summer research. Thank you to Rachel Burcin and John Dolan for organizing RISS and making this summer experience possible. Finally, thank you to the CMU Robotics Institute for funding this research.

REFERENCES

- [1] C. Moulin-Frier, S. M. Nguyen, and P. Oudeyer, "Self-organization of early vocal development in infants and machines: the role of intrinsic motivation," in *Frontiers in Psychology*, 2014.
- [2] R. Dubey, P. Agrawal, D. Pathak, T. L. Griffiths, and A. A. Efros, "Investigating human priors for playing video games," in *International Conference on Machine Learning (ICML)*, 2018.
- [3] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International Conference on Machine Learning (ICML)*, 2017.
- [4] J. Schmidhuber, "A possibility for implementing curiosity and boredom in model-building neural controllers," in *International Conference on Simulation of Adaptive Behavior (SAB)*, 1991.
- [5] R. Raileanu and T. Rocktäschel, "Ride: Rewarding impact-driven exploration for procedurally-generated environments," in *International Conference on Learning Representations (ICLR)*, 2020.
- [6] M. G. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos, "Unifying count-based exploration and intrinsic motivation," in *Advances in Neural Information Processing Systems (NIPS)*, 2016.
- [7] G. Ostrovski, M. G. Bellemare, A. van den Oord, and R. Munos, "Count-based exploration with neural density models," in *International Conference on Machine Learning (ICML)*, 2017.
- [8] Anonymous authors under review, "Interesting object, curious agent: Learning task-agnostic exploration," 2021.
- [9] A. L. Strehl and M. L. Littman, "An analysis of model-based interval estimation for markov decision processes," in *Journal of Computer and System Sciences (JCSS)*, 2008.
- [10] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. D. Turck, and P. Abbeel, "Vime: Variational information maximizing exploration," in *Advances in Neural Information Processing Systems (NIPS)*, 2016.
- [11] J. Schmidhuber, "Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts," in *Connection Science*, 2006.
- [12] A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. V. Hasselt, and D. Silver, "Successor features for transfer in reinforcement learning," in *International Conference on Neural Information Processing Systems (NeurIPS)*, 2017.
- [13] F. Fernández and M. Veloso, "Probabilistic policy reuse in a reinforcement learning agent," in *International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2006.
- [14] J. Hailu and G. Sommer, "On amount and quality of bias in reinforcement learning," in *International Conference on Systems, Man, and Cybernetics (SMC)*, 1999.
- [15] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell, "Policy distillation," in *International Conference on Learning Representations (ICLR)*, 2015.
- [16] S. Hansen, W. Dabney, A. Barreto, T. V. de Wiele, D. Warde-Farley, and V. Mnih, "Fast task inference with variational intrinsic successor features," in *International Conference on Learning Representations (ICLR)*, 2020.
- [17] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto, "Reinforcement learning with prototypical representations," in *International Conference on Machine Learning (ICML)*, 2021.
- [18] H. Tang, R. Houthooft, D. Foote, A. Stooke, O. X. Chen, Y. Duan, J. Schulman, F. DeTurck, and P. Abbeel, "#exploration: A study of count-based exploration for deep reinforcement learning," in *Advances in Neural Information Processing Systems (NIPS)*, 2017.
- [19] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Symposium on Theory of Computing*, 2002.
- [20] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. Chaplot, O. Maksymets, V. V. A. Gokaslan, S. Dharur, F. Meier, W. Galuba, A. Chang, Z. Kira, V. Koltun, J. Malik, and D. B. M. Savva, "Habitat 2.0: Training home assistants to rearrange their habitat," 2021, arXiv:2106.14405.
- [21] B. C. Stadie, S. Levine, and P. Abbeel, "Incentivizing exploration in reinforcement learning with deep predictive models," in *NIPS Workshop on Deep Reinforcement Learning*, 2015.
- [22] Y. W. Teh, V. Bapst, W. M. Czarnecki, J. Quan, R. H. J. Kirkpatrick, N. Heess, and R. P. Distral, "Robust multitask reinforcement learning," in *Neural Information Processing Systems (NeurIPS)*, 2017.
- [23] D. Pathak, D. Gandhi, and A. Gupta, "Self-supervised exploration via disagreement," in *International Conference on Machine Learning (ICML)*, 2019.

Multi-Modal Socially-Aware Imitation Learning for General Aviation

Jasmine Jerry Aloor¹, Jay Patrikar² and Sebastian Scherer²

Abstract—With an anticipated increase in aerial traffic in the near future in the form of crewed air-taxis, aircraft, and autonomous aerial systems, achieving safe human-operated and independent vehicle coordination in shared airspace is crucial. Current studies have investigated pedestrian and ground vehicle trajectories in multiagent environments in great detail. However, these studies have not yet analyzed aerial trajectories. There is a need to have safely separated aircraft in the vicinity of airports that behave as required when entering and leaving. This work takes a step forward to achieve safe manned-unmanned vehicle operations while learning from each other. A motion planning algorithm is developed using methods of behavior cloning using the inputs of state, past trajectory, and intent to determine the subsequent actions with a safety system using human intervention. We use real-world aircraft flight data as trajectories encoded as states, augmented with the agent's goal to generate optimal trajectories as future actions. We discuss an assured-safety intervention learning algorithm to expand the proposed approach to handle out-of-distribution compounding errors. Our system incorporates continuous learning allowing autonomous aircraft to integrate into regular manned traffic flow safely.

Index Terms—Motion and Path Planning, Imitation Learning, Social Navigation, Aerial Systems

I. INTRODUCTION

The future of aviation will see a massive increase in the integration of low-altitude autonomous flying aircraft, air taxis, personal air vehicles, and commercial urban aerial mobility (UAM) systems. In an unstructured environment such as one near non-towered airspace, this entails a high degree of inter-aircraft interaction to remain safe and reason about the potential intentions of each ‘agent.’ General Aviation (GA) consists of all non-military, non-commercial civilian flight operations and takes up the bulk of air traffic in uncontrolled airspace. GA pilots primarily use visual observations of the path of other aircraft to detect, sense, and avoid. Augmenting this with knowledge of aviation regulations, current weather, and experience, pilots make predictions of the path of other aircraft. Terminal Airspace describes the airspace surrounding an airport, where aircraft converge and also perform maneuvers such as descent and turns. Since 2018, the Federal Aviation Administration (FAA) has seen an increase in the number of GA flight hours and a resulting rise in GA accident rates [1]. Conflicts between aircraft are common, and resolution involves direct pilot-to-pilot communication. Traditional detect and avoid systems (DAA

¹Jasmine Jerry Aloor is with the Department of Aerospace Engineering, Indian Institute of Technology, Kharagpur, WB, India
jasminejerry@iitkgp.ac.in

²Jay Patrikar and Sebastian Scherer are with the Robotics Institute, School of Computer Science at Carnegie Mellon University, Pittsburgh, PA, USA {jpatrika, basti}@andrew.cmu.edu

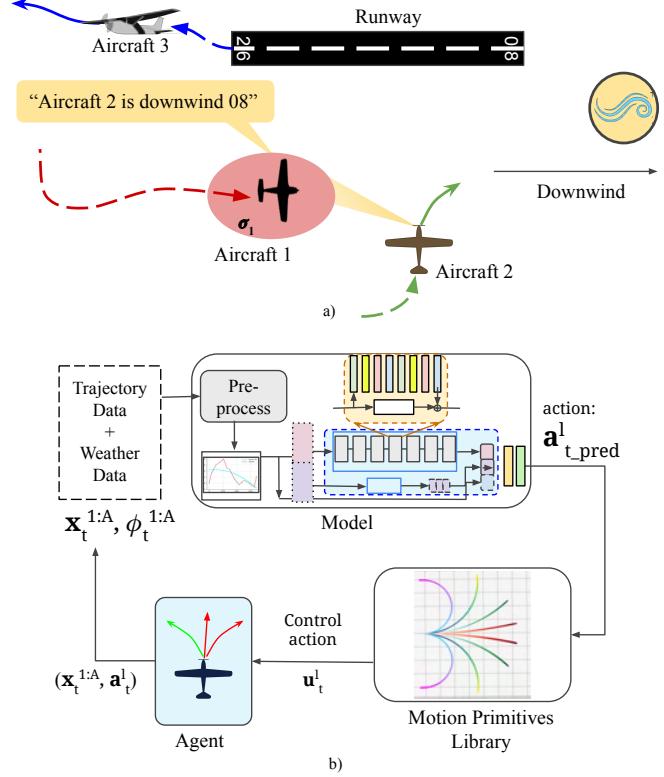


Fig. 1: Summary of the approach: a) The social navigation challenge for a pilot trying to land near a runway using a rectangular pattern in terminal airspace. b) Our solution involving imitation learning using trajectory and weather information to output the next action required to be taken.

such as TCAS or ACAS-X) are designed for the en-route phase of flight and try to prevent collisions rather than to reach goals and are unable to process multiple streams of information like vision and speech. Autonomous and crewed aircraft are strictly separated, which also limits flexibility and reduces the effectiveness of operations. In close vicinity of other vehicles or obstacles, they cannot react and avoid them to ensure safety.

Problem: Safe General Aviation Navigation: There is a need to have safely separated aircraft coordination, planning, and navigation in the absence of a centralized traffic control tower in the vicinity of airports. Pilots are expected to be socially compliant and follow the FAA guidelines to take actions that are acceptable within a social context. Following the basic rectangular traffic pattern reduces the possibility of conflicts at airports without an operating control tower. As the guidelines are not strictly enforced, there is scope for

flexibility in each trajectory while maintaining separation. With each aircraft having a specific goal, requiring to maintain safety and behave as expected when entering and leaving the formation, this becomes a social navigation problem.

Solution: To develop solutions to social navigation, two main directions have emerged in recent research: reinforcement learning (RL) [2] and inverse reinforcement learning (IRL) [3] based approaches. RL methods have a shortcoming when the reward functions structure is not present that can satisfy the requirements of navigation. Training in the real world is difficult and expensive with new policies, especially for safety-critical systems. Using a multi-agent simulator presents the issue of accurately modeling human-like piloting behavior for training a policy, which leads us to the initial problem. On the other hand, learning from expert human demonstrations using IRL and imitation learning (IL) can attempt to provide a natural, human-like motion.

Sequential decision-making problems using IL leverage the information of the next best action choice taken by an expert to learn a near-optimal policy more effectively than RL [4]. Imitation learning is a popular research direction that uses supervised learning for decision-making. Here, a policy is learned that imitates recorded behaviors, and the simplest form is known as behavior cloning. We present a novel trajectory planning method *name* that uses inputs of state, past trajectory, and intent to determine the next actions with a safety system using behavior cloning as our baseline algorithm.

We use real-world GA aircraft flight data as trajectories that are encoded as states, and the latent space augmented with the agent’s goal generates the next actions to be taken. We test the algorithm and learned policy to analyze the limitations of the behavior cloning approach. The results show that learning from demonstrations without interactive learning leads to poor results. By applying expert human intervention when the agent leaves a safe set, we aim to enable safety by repairing bad actions and re-positioning waypoints. Our approach has wide-ranging applications, from being a smart co-pilot for the regular crewed operation to allowing autonomous aircraft to safely integrate into regular manned traffic flow safely.

Contributions: The main contributions of this work are summarized as follows:

- 1) A curated set of dynamically feasible motion primitives frequently observed in GA aircraft near terminal airspace, generated and utilized for aircraft motion prediction.
- 2) A trajectory planning method is developed that uses information from agent-agent, agent-environment, and agent-context interactions to generate next actions.

The organization of the remainder of the paper is as follows: In Section II, we provide an overview of the methods used in aircraft motion prediction, social navigation, and imitation learning. In Section III, we introduce the approach, motion primitives generated, and our algorithm. In Section IV, we test our baseline model with selected metrics. In Sections

V and VI we outline future work using interventions and conclude respectively.

II. RELATED WORK

Aircraft motion prediction: As the number of air taxis, autonomous aircraft, and aerial mobility systems increases, there exists a need to have an efficient operation in shared airspace. Knowing the accurate trajectory models in the vicinity of terminal airspace is essential to develop advanced control technologies during close-proximity interactions. Previous works involved predicting motion by estimation of the aircraft’s state and using dynamic equations to propagate the estimate [5], combining different modes of operation to generate a trajectory [6] and learning probabilistic representations from historical data [7], [8]. Recent research has used Gaussian Mixture Models (GMMs) to learn aircraft trajectory positions and deviations from intended behavior in terminal airspace [9], [10].

Social navigation: Focusing on the prediction of a single aircraft’s trajectory would leave out the information of interaction with multiple aircraft and the effects on all trajectories. Socially compliant navigation that is focused on modeling human pedestrian behavior and motion has been extensively studied [11] where an S-LSTM model uses a ‘social pool’ layer connecting to every agent’s trajectory network to account for the inter-agent dynamics. Recent research [12], [13] applies this to predict multi-aircraft trajectories.

Imitation Learning: With an increase in the collection of expert demonstration data, many traditional imitation learning algorithms like behavior cloning have been developed policies that mimic human experts’ directly [3], or via feedback querying [14], [15]. DAGGER algorithm [16] is an online sampling framework that trains their current policy by collecting the correct action labels from the expert. As the expert actions are aggregated, the algorithm trains a new policy with significant improvement. However, it becomes inefficient when querying at every state visited by the algorithm that the expert would normally not visit, and the learner keeps all control with the expert receiving no feedback. HG-DAGGER algorithm [17] is an extension that allows selective querying of the expert where the novice policy is rolled out until it enters an unsafe region. The expert has full control of the system and returns it to a safe and stable state-space during which the expert action labels are collected. The recent Expert Intervention Learning (EIL) algorithm [18] learns the expert actions through demonstrations as well as the timing of the correction. The policy minimizes the querying of the expert over time and quickly learns the good and bad states, thereby reducing the need for interventions. However, these works do not provide a guarantee of the safety of the system. We build on these to develop a learning algorithm that makes decisions using expert data and discuss future directions to the work.

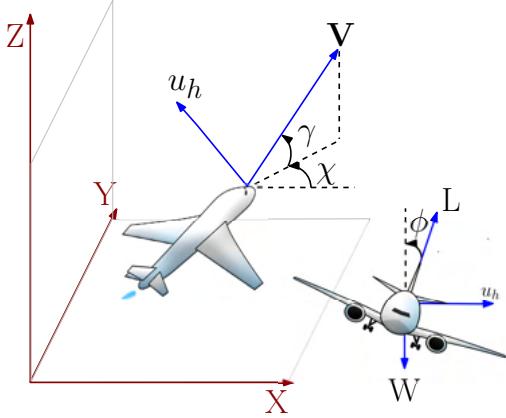


Fig. 2: Fixed wing 3D Coordinate frame

III. PROBLEM FORMULATION

A. Trajectory Data

This work uses the recently-released *TrajAir*¹ dataset, which is collected at the Pittsburgh-Butler Regional Airport (ICAO:KBTP) [19]. KBTP is a single runway GA airport located 10 miles North of the city of Pittsburgh, Pennsylvania. Non towered airports have an airport traffic pattern established for aircraft entering and leaving with specifications that include the direction and procedures. Airport traffic patterns are rectangular-shaped, developed to ensure that air traffic is flown into and out of an airport safely and in an orderly manner. KBTP has Left Traffic patterns for its runway, where all turns in the pattern are to the left. The dataset trajectories are smoothed using B-spline (basis-spline) approximate representation of order 2, an order k B-spline is formed by joining polynomials of degree $k - 1$.

B. Approach

We model the multi-agent system as a sequential decision making problem in continuous space, composed of A agents (vehicles) that interact over T time steps. At time t , let $\mathbf{x}_t^a = (x_t^a, y_t^a, z_t^a)$ and ϕ_t^a denote the position and weather context (taken as wind velocities) of the a^{th} agent. Let $\mathbf{x}_{t_1:t_2}^{1:A}, \phi_{t_1:t_2}^{1:A}$ and $\mathbf{a}_{t_1:t_2}^{1:A}$ denote the trajectories, context and actions over a $\{t_1, \dots, t_2\}$ time-horizon for all $\{1, \dots, A\}$ agents in that scene.

We formulate the problem as finding a distribution of future actions $\hat{\mathbf{a}}_{t_{obs}:t_{pred}+t_{obs}}^{1:A}$ conditioned on the past trajectories $\mathbf{x}_{1:t_{obs}}^{1:A}$ and context $\phi_{t:t_{obs}}^{1:A}$, where, t_{obs} is the observation time window and t_{pred} is the future time horizon (1).

$$\hat{\mathbf{a}}_{t_{obs}:t_{pred}+t_{obs}}^{1:A} \sim p(\hat{\mathbf{a}}_{t_{obs}:t_{pred}+t_{obs}}^{1:A} | \mathbf{x}_{1:t_{obs}}^{1:A}, \phi_{t:t_{obs}}^{1:A}) \quad (1)$$

The action space \mathcal{A} consists of a fixed library of 252 motion primitives represented by \mathbf{x}_t^l , where, $\mathbf{x}_t^l = (x_t^l, y_t^l, z_t^l)$ denote the positions of each trajectory in the library at time t (see Fig. 4). Each action \mathbf{a}_t^i is a pre-defined control and a sequence of resulting predicted trajectories $\mathbf{a}_{t_{obs}}^i = (\hat{\mathbf{u}}_{1:t_{obs}}^{1:A}, \hat{\mathbf{x}}_{1:t_{obs}}^{1:A})$.

¹<http://theairlab.org/trajair/>

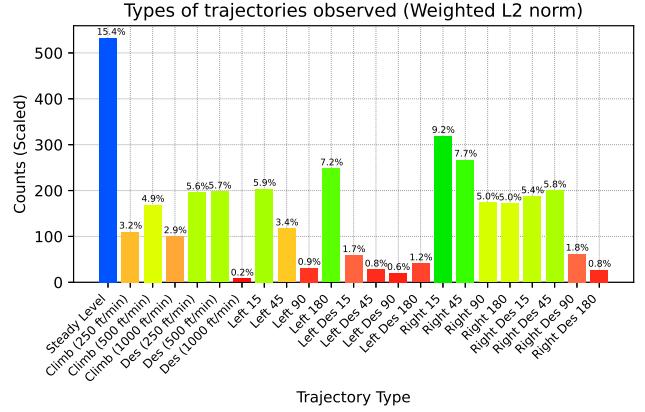


Fig. 3: Trajectory Classification: The broad categories of trajectories of the dataset along with climb rates (ft/min) and turning (heading angle °) are plotted

C. Motion primitives

We consider a fixed-wing aircraft flying in three-dimensional Euclidean space (Fig. 2). The kinematic equations (2) are used to calculate a set of trajectories in the inertial frame.

$$\dot{x} = v \cos \gamma \cos \chi \quad (2a)$$

$$\dot{y} = v \cos \gamma \sin \chi \quad (2b)$$

$$\dot{z} = v \sin \gamma \quad (2c)$$

$$\dot{\chi} = \frac{u_h}{v \cos \gamma} \quad (2d)$$

where, $(x, y, z) \in \mathbb{R}^3$, $v = \|\mathbf{V}\|$ is the inertial speed and the velocity vector \mathbf{V} is a function of the inertial speed (v), elevation angle (γ) and azimuth angle (χ). The lateral acceleration is represented by u_h , which is a function of the aircraft's bank angle, ϕ . We choose three broad categories of motion

- 1) Steady Level flight
- 2) Steady Climb and Descent
- 3) Coordinated Turns (Right, Left)

We vary the velocities from 60-120 knots in steps of 20 knots, we choose the rates of climb and descents (\dot{z}) to vary from $\{0, \pm 250, \pm 500, \pm 1000\}$ ft/min and set the bank angles as $\{2^\circ, 7^\circ, 15^\circ, 27^\circ\}$ to turn by $\{15^\circ, 45^\circ, 90^\circ, 180^\circ\}$ heading respectively over the chosen time-horizon. We classify the trajectories in the dataset into different climb rates based on the start and end heights ($\delta z / \delta t$). We then match the generated trajectories in the library with the trajectory dataset to understand the types of motions encountered using a weighted L2 Euclidean error distance over (x, y) points on the entire trajectory. We represent the minimum value and the matched library trajectories over all the scenes in $L(\mathbf{x}_{1:t_{obs}}^a)$

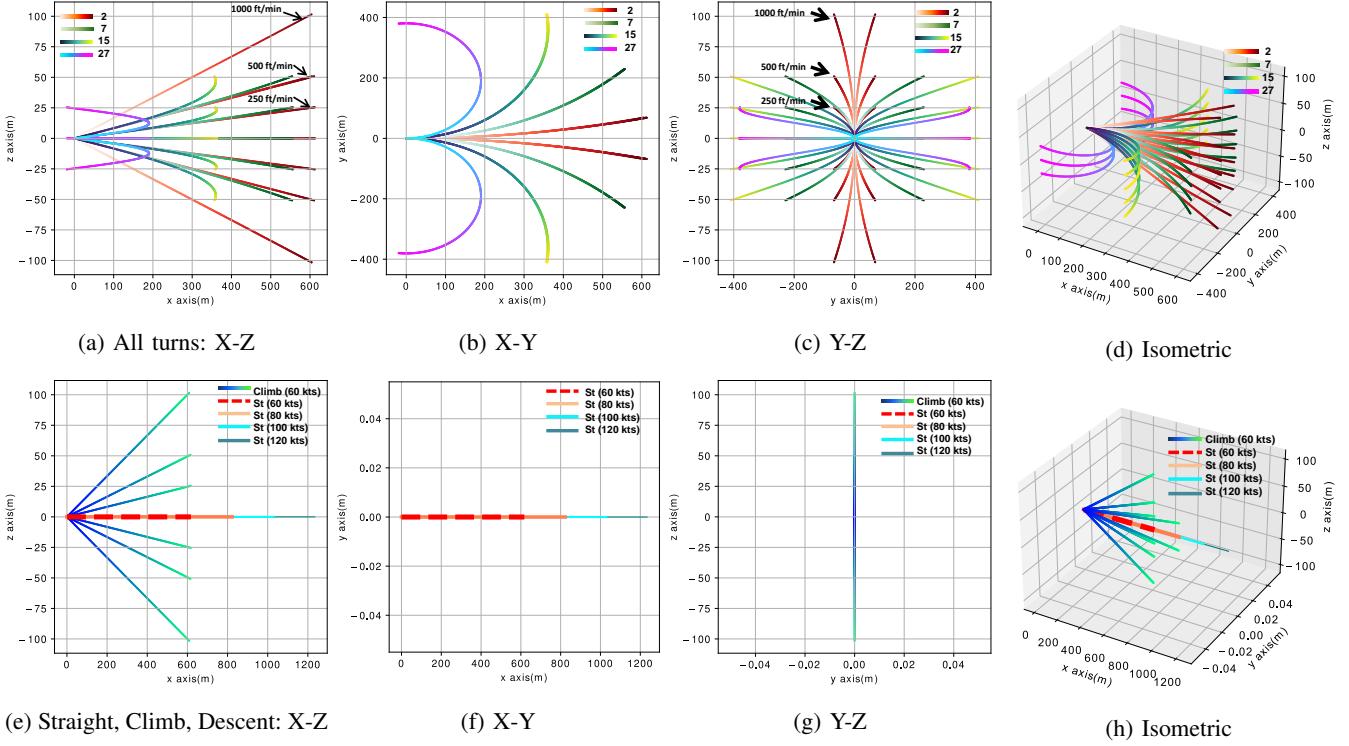


Fig. 4: Motion Primitives: The three broad categories of motion chosen are steady level flight, steady climb/descent, and coordinated turns. Here we show the X-Y, Y-Z, X-Z, and isometric views of the motion primitives all at 60 knots. Fig. 4e-4h also has steady level flight plotted for velocities 80, 100, and 120 knots

and $T(\mathbf{x}_{1:t_{obs}}^a)$ respectively.

$$\forall d \in \mathcal{D} \quad L(\mathbf{x}_{1:t_{obs}}^a) = \min_{\substack{l \in \{1 \dots \mathcal{N}\} \\ s.t. \dot{z}_t^l = \dot{z}_t^a}} \left(\sum_{i=t_1}^{t_2} \sum_{j \in (x,y)} (w_i [x_i^{l(j)} - x_i^{a(j)}])^2 \right)^{\frac{1}{2}} \quad (3a)$$

$$\forall d \in \mathcal{D} \quad T(\mathbf{x}_{1:t_{obs}}^a) = \arg \min_{\substack{l \in \{1 \dots \mathcal{N}\} \\ s.t. \dot{z}_t^l = \dot{z}_t^a}} \left(\sum_{i=t_1}^{t_2} \sum_{j \in (x,y)} (w_i [x_i^{l(j)} - x_i^{a(j)}])^2 \right)^{\frac{1}{2}} \quad (3b)$$

where, $(x_t^{l(x)}, x_t^{l(y)}) = (x_t^l, y_t^l)$, w_i is the weights applied to the segments, \mathcal{N} is the total number of trajectories in the library and \mathcal{D} is the number of scenes in the dataset. The preliminary analysis showed the most frequent types of motions observed were left and right turns, which is expected from the traffic pattern to be followed near KBTP. We also observe a low number of steady steep climb and descent rates and no sharp turns with a steep climb and descent rates.

D. Model Details

We propose a behavior cloning algorithm that takes as input the past trajectories of all the agents, along with the weather context, to predict their possible action distribution. The network uses a Temporal Convolutional Network (TCN) to process the sequential trajectory data. TCN layers encode a trajectory's spatio-temporal information into a latent vector without losing the underlying data's temporal (causal)

relations in the underlying data [20]. We use TCNs as an alternative to using LSTMs [12] for encoding the trajectories.

We break the trajectories in a scene into sequences of length $t_{obs} + t_{pred}$ with a certain minimum number of agents constant across each sequence. The number of agents can change from sequence to sequence. For each agent in a given scene, the raw trajectory in absolute coordinates is encoded using the same TCN layers.

$$h_{obs}^a = TCN_{obs}(\mathbf{x}_{1:t_{obs}}^a) \quad \forall a \in \{1, \dots, A\} \quad (4)$$

where, h_{obs}^a is the encoded vector of agent a . The encoded information of the other agents is max-pooled to get a set of representations of other agents with the greatest numerical value.

$$\bar{h}_{obs}^a = \forall b \in \{1, \dots, A\} \neq a \maxpool\{h_{obs}^b\} \quad (5)$$

To include the environmental context, the raw context vector for each agent is encoded using a standard CNN layer, and the output is concatenated to the TCN encoded trajectories.

$$h_{enc}^a = h_{obs}^a \oplus \bar{h}_{obs}^a \oplus CNN(\phi_{1:t_{obs}}^a) \quad \forall a \in \{1, \dots, A\} \quad (6)$$

In the goal-conditioned network, we append the agent's goal as a one-hot vector representation of the final goal of a particular agent as the eight cardinal directions along with two runway ends. Finally, the output is passed through a Fully Connected layer to get the correct dimension for the predictions from the motion primitive library.

$$a_{t_{obs}:t_{obs}+t_{pred}}^l = MLP(h_{enc}^a) \quad (7)$$

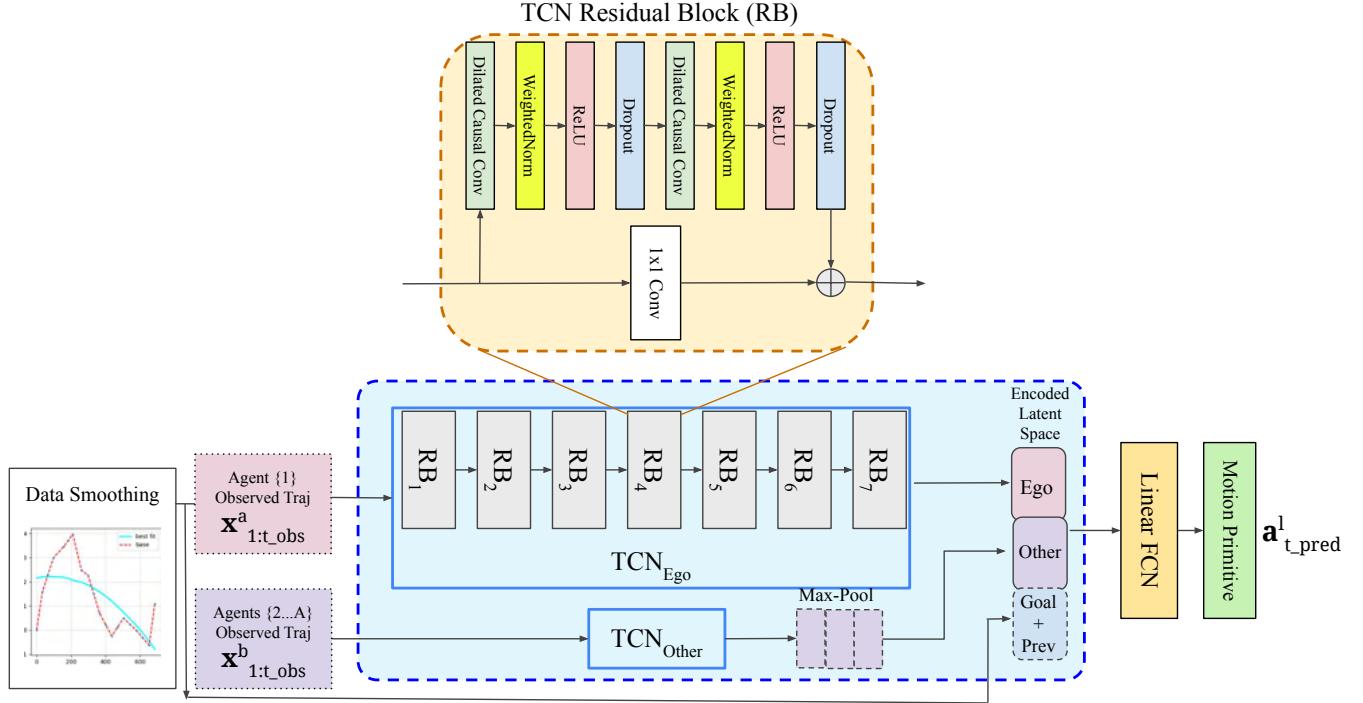


Fig. 5: Overview of the model: The smoothed trajectories of all agents are passed into the TCN separately. For each agent, the encoded space of all other agents is concatenated after a max pool. If the network inputs the agent's goal (g^a), then it is appended to the encoded variable along with the previous iteration's prediction.

The \mathcal{L}_{traj} measures how close the predicted trajectory is to the ground-truth trajectory using an L2 Euclidean distance loss.

$$\mathcal{L}_{traj} = L2(\mathbf{x}_{t_{obs}:t_{obs}+t_{pred}}^a, \hat{\mathbf{x}}_{t_{obs}:t_{obs}+t_{pred}}^a) \quad (8)$$

The \mathcal{L}_{act} measures how close the predicted action is to the pre-processed matched action using a Cross-Entropy Loss (CEL).

$$\mathcal{L}_{act} = CEL(\hat{\mathbf{a}}_{t_{obs}:t_{pred}+t_{obs}}^a, T(\mathbf{x}_{1:t_{obs}}^a)) \quad (9)$$

For training, we use the Adam optimizer with a learning rate of $1e - 4$.

IV. EVALUATIONS AND DISCUSSION

We evaluate the trajectories on one set of 7 consecutive days of data portion in the TrajAir dataset. To accurately capture the dynamics of a particular motion in x-y-z coordinates, we use $t_{obs} = 20$ sec and $t_{pred} = 20$ sec. We select the action predicted from the trajectory library at $t_{obs} + 1$ and match it with the trajectory followed by the agent in that scene.

The evaluation metrics used are

- 1) Average Displacement Error (ADE): L2 Euclidean distance over the entire trajectory
- 2) Weighted Displacement Error (WADE): Weighted end-point L2 Euclidean distance over the whole trajectory.
- 3) Cross entropy loss (CEL): For the selection of right actions from the library.

We use results for the best of $N = 5$ queries to the network to record all the metrics (ADE/FDE/CEL scores).

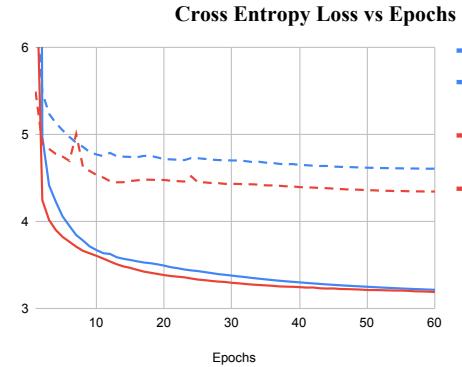
Metrics	ADE (m)	FDE (m)	CEL
Best Case	13.60	27.43	2.64
Failure Case	18.30	31.72	3.33

TABLE I: Evaluation metrics for the best case and failure case observed in the test data.

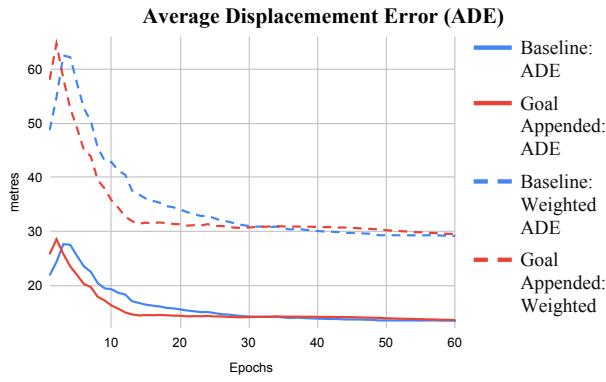
Figure 7 shows the qualitative results for a right turn scenario along with the X-Plane visualization. The x-y tracked trajectory is shown on the top right, which is a segment from the rectangular pattern followed by the agent in the dataset. Figure 8 shows a failure case where the predicted actions were in the opposite direction to the motion of the agent, thus leading to an incorrect path. Table I shows the quantitative results for the behavior cloning network.

V. FUTURE WORKS

Behavior cloning is known to require a large number of expert demonstrations and, at the same time, deviates significantly when encountered with out-of-distribution states [3], [21]. A formulation similar to Inverse Reinforcement Learning uses generative adversarial training to fit distributions of states and actions. The algorithm, Generative Adversarial Imitation Learning (GAIL) [22] learns policies directly from data. We aim to incorporate GAIL-like methods in our training along with goal conditioning as our next step.



(a) Cross entropy loss (CEL)



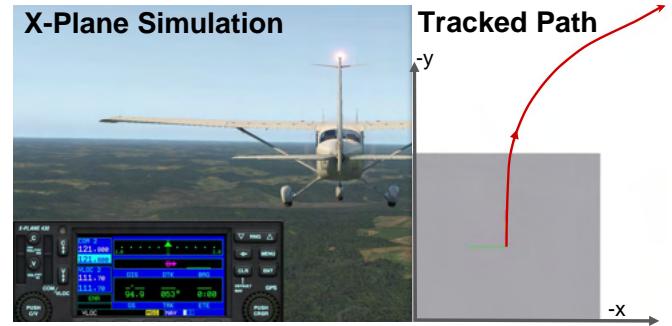
(b) Average and Weighted Average Displacement Error

Fig. 6: Evaluation metrics

An aircraft pilot has multiple directives needed to follow to navigate safely in an airspace. We seek to determine all constraints and encode them appropriately to be included while training our model. Approaches that can collect feedback interactively by querying the expert have been explored [14], but can be impractical due to the algorithm querying in states that are not natural for the human expert. Additionally, the lack of full control to the expert causes a delayed response. Another method is to provide the human expert full control and the option to intervene when required, which is more natural and provides the algorithm the additional information of the ‘bad’ states to avoid and technique to recover from it [17], [18]. We plan to build on this method to use interventions from a supervisor to handle achieve safe manned-unmanned vehicle teaming to improve the system performance and have each teammate learn from each other in different aircraft operations. When our agent goes out of the safe zone, we will use the expert intervention to bring it back to safety and update the learner algorithm.

VI. CONCLUSION

This work presents a model using imitation learning for socially aware navigation of dynamical systems in the general aviation domain. It also offers a curated set of motion primitives for aircraft that are frequently observed



Visualization of dataset trajectory and motion primitives selected

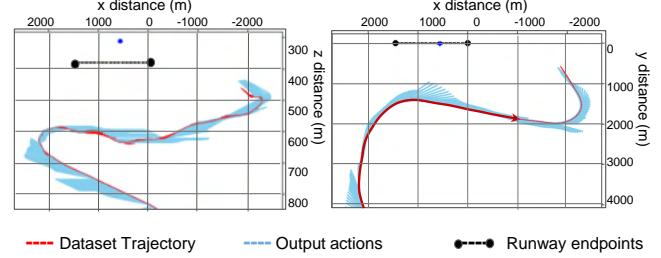


Fig. 7: Actions input to the X-Plane visualizer. Top right: The tracked trajectory. Bottom: The dataset trajectory and predicted actions

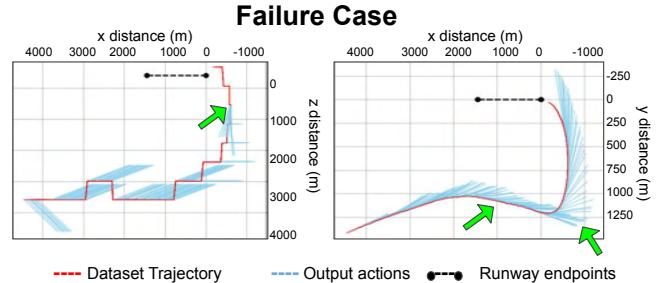


Fig. 8: The green arrows show locations where actions were wrongly predicted: opposite to the direction of motion of the agent.

in general aviation navigation. The current results for behavior cloning are good given the inputs of trajectory and weather information. It is, however, the first step in the aviation domain for social robotics and automation. There is a great scope for improvement using interactive learning and intervention-based methods that will be compared to our proposed method.

ACKNOWLEDGMENT

This work was sponsored by the AirLab at the Robotics Institute, Carnegie Mellon University, as a part of the Robotics Institute Summer Scholars (RISS) program. We also give our special thanks to Ms. Rachel Burcin, Dr. John Dolan, and Ms. Jennie Piotrkowski for their support throughout the program.

REFERENCES

- [1] "Preliminary aviation statistics," USA National Transportation Safety Board, 2000-2019. [Online]. Available: http://www.ntsb.gov/investigations/data/pages/aviation_stats.aspx
- [2] M. Everett, Y. F. Chen, and J. P. How, "Collision avoidance in pedestrian-rich environments with deep reinforcement learning," *IEEE Access*, vol. 9, pp. 10357–10377, 2021.
- [3] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.
- [4] W. Sun, A. Venkatraman, G. J. Gordon, B. Boots, and J. A. Bagnell, "Deeply aggravated: Differentiable imitation learning for sequential prediction," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3309–3318.
- [5] G. Chatterji, "Short-term trajectory prediction methods," in *Guidance, Navigation, and Control Conference and Exhibit*, 1999, p. 4233.
- [6] R. Slattery and Y. Zhao, "Trajectory synthesis for air traffic automation," *Journal of guidance, control, and dynamics*, vol. 20, no. 2, pp. 232–238, 1997.
- [7] M. J. Kochenderfer, L. P. Espindle, J. K. Kuchar, and J. D. Griffith, "A comprehensive aircraft encounter model of the national airspace system," *Lincoln Laboratory Journal*, vol. 17, no. 2, pp. 41–53, 2008.
- [8] C. Lowe and J. P. How, "Learning and predicting pilot behavior in uncontrolled airspace," in *AIAA Infotech@ Aerospace*, 2015, p. 1199.
- [9] S. T. Barratt, M. J. Kochenderfer, and S. P. Boyd, "Learning probabilistic trajectory models of aircraft in terminal airspace from position data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3536–3545, 2018.
- [10] S. Jung and M. J. Kochenderfer, "Learning terminal airspace traffic models from flight tracks and procedures," in *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*. IEEE, 2019, pp. 1–8.
- [11] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.
- [12] Z. Zhao, W. Zeng, Z. Quan, M. Chen, and Z. Yang, "Aircraft trajectory prediction using deep long short-term memory networks," in *CICTP 2019*, 2019, pp. 124–135.
- [13] Z. Xu, W. Zeng, X. Chu, and P. Cao, "Multi-aircraft trajectory collaborative prediction based on social long short-term memory network," *Aerospace*, vol. 8, no. 4, p. 115, 2021.
- [14] S. Ross and D. Bagnell, "Efficient reductions for imitation learning," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 661–668.
- [15] S. Ross, G. J. Gordon, and J. A. Bagnell, "No-regret reductions for imitation learning and structured prediction," in *In AISTATS*. Citeseer, 2011.
- [16] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.
- [17] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, "Hg-dagger: Interactive imitation learning with human experts," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8077–8083.
- [18] J. Spencer, S. Choudhury, M. Barnes, M. Schmittie, M. Chiang, P. Ramadge, and S. Srinivasa, "Learning from interventions," in *Robotics: Science and Systems (RSS)*, 2020.
- [19] J. Patrikar, B. Moon, S. Ghosh, J. Oh, and S. Scherer, "Trajair: A general aviation trajectory dataset," Jun 2021. [Online]. Available: <https://doi.org/10.1184/R1/14866251.v1>
- [20] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.
- [21] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *arXiv preprint arXiv:1811.06711*, 2018.
- [22] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, pp. 4565–4573, 2016.

Class-Imbalanced Learning via Bilevel-Optimized Weight Decay

Shaden N Alshammary¹, Yu-Xiong Wang², Deva Ramanan³, Shu Kong³

Abstract—Real-world data tends to follow long-tailed distributions: there are some common classes that have abundant data while many rare classes that have scarce data. However, performance on rare classes is often crucial in many real-world applications. This motivates the well-studied class-imbalanced learning (CIL) problem, which aims for high accuracy averaged over the imbalanced classes. In CIL, a naively trained model performs well on common classes but poorly on rare classes. Popular solutions for emphasizing rare classes are based on data resampling or loss reweighting. We explore the orthogonal direction by balancing network weights via regularization for CIL. Our motivation is based on the observation that a naively trained model has imbalanced classifier weights w.r.t norms. We first revisit weight decay, a popular regularizer adopted in deep learning, and find that carefully tuning it notably improves CIL, rivaling prior arts. Further, to avoid manually tuning weight decay, we exploit bilevel optimization that automatically tunes it during training. This allows for learning with per-layer weight decays, achieving even better performance. Our experiments on a popular CIL dataset show that learning with bilevel optimization of weight decay achieves the state-of-the-art performance.

I. INTRODUCTION

Real-world data often follows long-tailed distributions, e.g., class labels are imbalanced such that a few *common-classes* have abundant data while many *rare-classes* have scarce data [1]–[3]. However, high recognition accuracy on rare-classes is often crucial in various real-world applications (e.g., cancer cell recognition and malicious internet content recognition) [4]. This motivates the well-studied class-imbalanced learning (CIL) [5]–[7], which evaluates classification accuracy macro-averaged over all classes yet trains the model over class-imbalanced training set.

Status quo. In CIL, a naively trained model performs poorly on rare-classes because of scarce data in the rare-classes [5], [8]. To emphasize the accuracy on rare-classes [9], existing methods commonly adopt loss reweighting or data resampling during training [2], [7], [10]–[12]. While it makes sense to balance classes’ distribution or their impact on training losses, doing so does not exploit training examples properly. For example, oversampling rare-class examples artificially form training batches consisting more such data, as a result the trained model overfits to them and generalizes poorly [5], [12], [13]. On the contrary, recent work shows that decoupling feature learning and classifier learning performs much better [14] – first learning deep features without using any class balancing techniques, and then freezing the features and learning the classifier using some balancing techniques.

Other methods use transfer learning techniques that learns features from common classes and repurpose them for rare-class recognition [5], [15].

Motivation. Empirically, for CIL, a naively trained model has imbalanced norms of classifier weights, as shown in Figure 1. This motivates our CIL method that is to balance network weights via regularizing their norms norms. Moreover, this observation is also noted in [6] which shows that post-hoc normalization of classifiers boosts performance. Hypothetically, post-hoc normalizing only the classifier weights is insufficient for CIL because other layers are also biased towards common classes in an implicit way. Therefore, in this work, we propose an algorithm to regularize per-layer weight decay to better regularize network weights.

Contribution. We propose to balance network weights w.r.t norms through regularization in the training of the network. One common regularizer is weight decay, as known as the L_2 norm penalty. Weight decay encourages to learn small weights and hence serves to balance network weights. Interestingly, weight decay is underexplored in CIL. We revisit weight decay and conclude that tuning it boosts CIL performance. Furthermore, because tuning weight decay is nontrivial that requires manual efforts, we adopt bilevel optimization to automatically tune weight decay. Particularly, we propose a simple Population-based Bilevel Optimization (PBO) approach to efficiently optimize weight decay parameters. Our PBO allows training with *per-layer weight decay* that achieves the state-of-the-art on a popular CIL benchmark.

II. RELATED WORK

Class-Imbalanced Learning. Natural data tends to follow long-tailed or imbalanced distributions. However, performance on low-data regime is crucial in many real-world tasks [2], [5], [16]. This motivates the well-studied class-imbalanced learning (CIL) problem, which equally evaluates performance on imbalanced classes. Many CIL methods emphasize rare-class performance through loss reweighting and data resampling [5], [10], [12]. Specifically, data resampling focuses on modifying the training data to make it more class balanced such as under-sampling common-classes [17] and over-sampling rare-classes [12], [18]. Loss reweighting focuses on emphasizing rare-classes in the loss function [5], [7], [10], [19]. *Transfer learning* methods transfer knowledge learned on the common-classes to rare-classes [20], [21]. Recent work examines the training procedure and find CIL to be better addressed by decoupling feature learning and classifier learning, rather than jointly training them [14], [22]. [23] find that the SGD momentum causes issues in CIL

¹ Massachusetts Institute of Technology

² University of Illinois Urbana-Champaign

³ Robotics Institute, Carnegie Mellon University

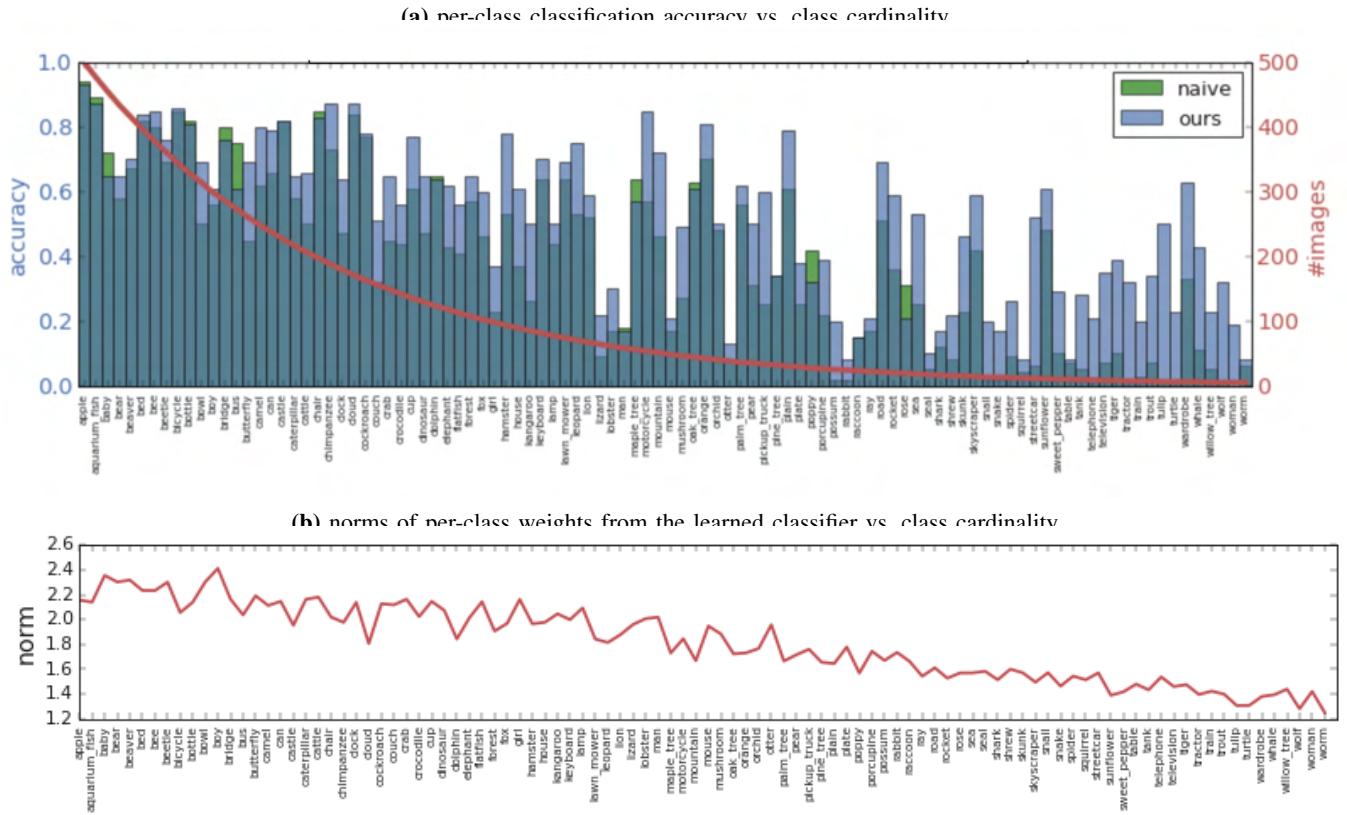


Fig. 1: Class-imbalanced Learning (CIL) trains a K -way classification network on class-imbalanced data, as shown by red curve in (a). A naively trained model performs well on common-classes but poorly on rare-classes, as shown by the green bars in (a). (b) A naively trained model has imbalanced classifier weights w.r.t norms: common/rare classes have large/small weights. This motivates us to balance network weights during training for CIL. Our method regularizes network training motivated to balance network weights. By doing so, “ours” improves long-tailed recognition for all classes, as shown in blue bars in (a). (The figures are based on results on the long-tailed version of CIFAR100 dataset with imbalance factor 100.)

that prevent further improvement. We explore the orthogonal direction of *parameter regularization*.

Regularization. For better generalization and alleviating overfitting, regularization plays a crucial role in optimization and learning [24]. There exist many regularization methods [25], [26], such as weight normalization [27], [28], early-stopping [29], data augmentation [30], dropout [31], etc. Among various constraints, weight decay, or L2-normalization penalty on the parameters, is a common and effective technique [32], [33]. In this work, we specifically explore weight decay which encourages to learn small weights to alleviate overfitting. This serves our intuition to balance network weights in CIL. Furthermore, we explore bilevel optimization methods to automatically tune weight decay during training. This makes our approach distinct from the common use of weight decay. We find learning with bilevel optimized weight decay to outperform existing CIL methods notably.

Hyperparameter Tuning and Bilevel Optimization. Tuning hyperparameters is crucial to alleviate overfitting and hence to learn more generalizable models. But hyperparameter tuning is non-trivial in terms of striving for an optimal balance of all regularizations [34]. It is common to assume

independent hyperparameters, e.g., only tuning weight decay by fixing learning rates, batch size, augmentation, etc. Doing so simplifies tuning: one can train models using each of some predefined hyperparameter values, and select the best-performing checkpoint via validation [35]. Such a tuning method requires manual efforts and is sub-optimal. Therefore, some work propose to use bilevel optimization to optimize hyperparameters along with learning model parameters. The typical criterion of optimizing hyperparameters is to maximize the performance on the validation set [36], [37]. There are different types of bilevel optimization on hyperparameter, such as SGD-based methods [37], [38] and model-free search based methods [35], [39]. Empirically, we find that the former does not work, presumably because of the difficulty of bilevel optimizing between low-dimensional hyperparameters and high-dimensional network parameters [37]. On the contrary, the latter works well. In this work, we adopt model-free bilevel optimization and contribute a new algorithm.

III. CLASS-IMBALANCED LEARNING WITH BILEVEL OPTIMIZATION OF WEIGHT DECAY

Problem Formulation. Given a training set of class-imbalanced data from K classes, Class-Imbalanced Learning

(CIL) learns a K -way classifier and measures accuracy macro averaged over all the K classes.

We are interested in training a deep neural network for K -way classification. We denote the network as $f(\cdot; \mathbf{Q})$ with M layers parameterized by $\mathbf{Q} = \{\theta_l\}_{l=1}^M$, where θ_l is the weights of layer- l . For a data example x , the network outputs $\hat{y} = f(x; \mathbf{Q}) \in \mathbb{R}^K$, a K -dimensional score vector. We measure the difference between the \hat{y} and the ground-truth one-hot y using a loss function $\ell(\hat{y}, y)$, e.g., the cross-entropy loss as used in this work. Denote $\mathcal{U} = \{(x_i, y_i)\}_{i=1}^N$ as the whole training set that contains N examples, we minimize the following to train $f(\cdot; \mathbf{Q})$:

$$\mathcal{L}_{\text{error}}(\mathbf{Q}) = \frac{1}{|\mathcal{U}|} \sum_{(x_i, y_i) \in \mathcal{U}} \ell(f(x_i; \mathbf{Q}), y_i) \quad (1)$$

By learning $f(\cdot; \mathbf{Q})$ on the training set \mathcal{U} , we hope it generalizes well to unseen testing examples. That said, instead of selecting a model that achieves the best performance on the training set, we commonly use a validation set \mathcal{V} to select the best-performing checked point.

Weight Regularization. The model $f(\cdot; \mathbf{Q})$ trained by minimizing (1) might not generalize well due to overfitting to the training data. To alleviate overfitting, we adopt regularization techniques such as weight decay. Furthermore, for CIL, a network trained to minimize Eq. (1) has imbalanced weights, as shown in Figure 1: classifiers have larger weights on common classes w.r.t norms while smaller weights on rare classes. This is because common classes have more data that dominate the training thus yield large weights. We are motivated balance network weights for CIL. Particularly, we exploit the weight decay, as known as L_2 -norm regularizer. Weight decay encourages learning small weights, and is a common regularizer in deep learning. However, it is underexplored in CIL – the literature of CIL simply fixes weight decay to some values without tuning [40], [41] for paper [42], [43]. We write weight decay penalty as below:

$$\mathcal{L}_{\text{wd}}(\mathbf{Q}, \boldsymbol{\lambda}) = \sum_{l=1}^M \lambda_l \|\theta_l\|_2^2, \quad (2)$$

where λ_l is the hyperparameter of weight decay on the weight at layer- l , and $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \dots, \lambda_M]$. We train the network $f(\cdot; \mathbf{Q})$ by optimizing \mathbf{Q} to minimize $\mathcal{L}^{\mathcal{U}} = \mathcal{L}_{\text{error}} + \mathcal{L}_{\text{wd}}$ over the training set \mathcal{U}

$$\mathbf{Q}^*(\boldsymbol{\lambda}) = \operatorname{argmin}_{\mathbf{Q}} \left\{ \mathcal{L}^{\mathcal{U}}(\mathbf{Q}, \boldsymbol{\lambda}) \equiv \mathcal{L}_{\text{error}}(\mathbf{Q}) + \mathcal{L}_{\text{wd}}(\mathbf{Q}, \boldsymbol{\lambda}) \right\} \quad (3)$$

Here we write network parameters \mathbf{Q} as a function over weight decay hyperparameter $\boldsymbol{\lambda}$. Typically, we use a single λ to regularize the whole network parameters without varying across layers, i.e., $\lambda_1 = \lambda_2 = \dots = \lambda_M = \lambda$. In our work, we find that carefully tuning λ via validation drastically improves CIL, as convincingly shown in Figure 3. However, hand-tuning requires repeatedly training independent models and selecting the best-performing model via validation. Such costly manual efforts prevent using fine granularity of weight

Algorithm 1: SGD-Based Bilevel Optimization

```

initialize base model  $\mathbf{Q}^{(0)}$ ;
initialize base weight decay parameter  $\boldsymbol{\lambda}^{(0)}$  ;
learning rates  $\alpha_{\mathbf{Q}}$  and  $\alpha_{\boldsymbol{\lambda}}$ 
for iteration  $t = 1, 2, \dots, T$  do
    sample a batch of training data, denoted as  $\mathbf{B}$ ;
    Compute  $\mathcal{L}^{\mathcal{U}}(\mathbf{Q}^{(t-1)}, \boldsymbol{\lambda}^{(t-1)}; \mathbf{B})$  according to Eq.(3)
     $\mathbf{Q}(\boldsymbol{\lambda}^{(t-1)}) = \mathbf{Q}^{(t-1)} - \alpha_{\mathbf{Q}} \nabla \mathcal{L}^{\mathcal{U}}(\mathbf{Q}; \boldsymbol{\lambda}^{(t-1)}, \mathbf{B})$ 
    sample a batch of validation data, denoted as  $\mathbf{G}$ ;
    Compute the criterion  $\mathcal{L}^{\mathcal{V}}$  (e.g., accuracy) on the val-set.
     $\boldsymbol{\lambda}^{(t)} = \boldsymbol{\lambda}^{(t-1)} - \alpha_{\boldsymbol{\lambda}} \nabla_{\boldsymbol{\lambda}} \mathcal{L}^{\mathcal{V}}(\mathbf{Q}(\boldsymbol{\lambda}^{(t-1)}); \mathbf{G})$ 
    Use  $\boldsymbol{\lambda}(t)$  to compute  $\mathcal{L}^{\mathcal{U}}(\mathbf{Q}^{(t-1)}, \boldsymbol{\lambda}^{(t)})$  and learn  $\mathbf{Q}^{(t)}$ 
end

```

decays, e.g., per-layer weight decay, which is expected to improve further. Intuitively, different layers need different degree of regularizations: compared to low-level generic layers, high-level layers are class-specific and hence need stronger regularization. Therefore, we explore methods to automatically tune weight decay for each layer.

A. Bilevel Optimization of Weight Decay for Class Imbalanced Learning

Recall that we typically tune hyperparameters via validation: we train independent models by setting λ as one of candidate values, and select the “best-performing” checkpoint with a validation set \mathcal{V} measured by a criterion $\mathcal{L}^{\mathcal{V}}$, e.g., classification accuracy macro averaged over all classes. This is essentially a bilevel optimization problem that has two loops: the inner loop is training network parameters by minimizing errors on the training data, and the outer loop optimizes the hyperparameters by minimizing errors on the validation set. A generic bilevel optimization performs the two levels of optimization more frequently as below:

$$\begin{aligned} \boldsymbol{\lambda}^* &= \operatorname{argmin}_{\boldsymbol{\lambda}} \mathcal{L}^{\mathcal{V}}(\mathbf{Q}^*(\boldsymbol{\lambda})) \\ \text{subject to } \mathbf{Q}^*(\boldsymbol{\lambda}) &= \operatorname{argmin}_{\mathbf{Q}} \mathcal{L}^{\mathcal{U}}(\mathbf{Q}) \end{aligned} \quad (4)$$

Gradient Descent based Bilevel Optimization. One straightforward approach is based on gradient descent [37]. Algorithm 1 depicts the pseudo code of this method. While the idea is simple, the gradient descent based method does not work empirically, presumably because of the difficulty of bilevel optimization between low-dimensional hyperparameters $\boldsymbol{\lambda}$ and high-dimensional network parameters \mathbf{Q} [37].

Model-Free Bilevel Optimization. Model-free approaches include grid search and random search [35]. Empirically, it is trivial to parallelize model-free methods over computing resources and they tend to perform well in practice [44], [45]. We adopt model-free bilevel optimization in this work. Next, we elaborate our population-based bilevel optimization method.

B. Population-based Bilevel Optimization (PBO)

We propose a **Population-based Bilevel Optimization (PBO)** method which is model-free. The essential idea is

Algorithm 2: Population-based Bilevel Optimization of a single λ

```

initialize base model  $\mathbf{Q}^{(0)}$ ;
initialize base weight decay parameters  $\lambda$ 
define  $K$  candidate hyperparameters  $\{\alpha_0, \dots, \alpha_K\}$ 
for iteration  $t = 1, 2, \dots, T$  do
    // train with different  $\lambda$  values
    for  $k = 1, \dots, K$  do
         $\lambda \leftarrow \alpha_k$ ;
         $\mathbf{Q}_k \leftarrow \underset{\mathbf{Q}}{\operatorname{argmin}} \mathcal{L}_{\text{error}}(\mathbf{Q}^{(t-1)}; \mathcal{U}) + \lambda \sum_{l=1}^M \|\theta_l\|_2^2$ ;
    end
    // select the best-performing model on
    // the validation set
     $k^* \leftarrow \underset{k}{\operatorname{argmin}} \mathcal{L}^V(\mathbf{Q}_k; \mathcal{V})$ 
     $\mathbf{Q}^{(t)} \leftarrow \mathbf{Q}_{k^*}$ 
end

```

Algorithm 3: Population-based Bilevel Optimization of Per-Layer Weight Decay

```

initialize base model  $\mathbf{Q}^{(0)}$ ;
initialize weight decay  $\lambda = [\lambda_1, \dots, \lambda_M]$ 
define  $K$  candidate hyperparameters  $\{\alpha_0, \dots, \alpha_K\}$ 
for iteration  $t = 1, 2, \dots, T$  do
    select layer  $j \in [1, M]$ 
    for  $k = 1, \dots, K$  do
         $\lambda_l \leftarrow \alpha_k$ ;
         $\mathbf{Q}_{(k)} \leftarrow \underset{\mathbf{Q}}{\operatorname{argmin}} \mathcal{L}_{\text{error}}(\mathbf{Q}^{(t-1)}) + \sum_{l=1}^M \lambda_l \|\theta_l\|_2^2$ ;
    end
    // select the best-performing model on
    // the validation set
     $k^* \leftarrow \underset{k}{\operatorname{argmin}} \mathcal{L}^V(\mathbf{Q}_k; \mathcal{V})$ 
     $\mathbf{Q}^{(t)} \leftarrow \mathbf{Q}_{k^*}$ 
end

```

that, we (1) train a *population* of K models simultaneously using K different hyperparameters $\lambda \in \{\alpha_1, \dots, \alpha_K\}$, (2) periodically evaluate K checkpoints, (3) select the best-performing one on the validation set and use it to reinitialize all the K models that are being trained, (4) keep training with the K different λ 's and repeat the above till the end of training. Our method builds on a hypothesis that searching for better and better weight decays during the whole course of training yields better performance in the end. Our method is conceptually similar to [46], which however only reinitializes underperforming models; in contrast, our PBO method always picks the best-performing checked point to initialize all other models and continue training them. In Algorithm 2, we illustrate our PBO algorithm for bilevel optimizing a single weight decay hyperparameter λ . Extending the algorithm to bilevel optimize per-layer weight decay is straightforward, as described in Algorithm 3.

While the algorithms are simple to understand, we find that, with a constant learning rate or a single learning rate scheduler, our PBO method does not achieve better performance; however, using cyclic learning rate schedule [47] make PBO shine. Our observation matches the number reported in [48]. Intuitively, cyclic learning rates help find better local minima and hence better performance in the end. We carry out ablation study on this point in the experiment section.

IV. EXPERIMENTS

We carry out experiments to validate our Population-based Bilevel Optimization (PBO) method for class-imbalanced learning (CIL). First, we show the importance of tuning weight decay for CIL. Second, we study the significance of using cyclic learning rate in bilevel optimization of weight decay with our PBO method. Third, we study PBO in learning with per-layer weight decays. Finally, we benchmark our PBO and compare against existing CIL methods. We start with experiment setup.

Dataset and Evaluation Metric. We conduct experiments using the long-tailed CIFAR100 dataset (imbalance factor as

100) [49]. The dataset is a modified version of the original CIFAR100 by reducing the number of training examples per class such that it follows an exponential distribution. Each of the training and validation datasets has 100 category classes. Following the literature [49], we use the class-balanced CIFAR100 validation set for evaluation; this does not affect evaluation because we macro-average per-class accuracy as the evaluation metric.

Implementation. We train a ResNet34 network [50] using the cross-entropy loss (CE), SGD optimizer with momentum 0.9. We set batch size as 64 for all of the experiments. We use the PyTorch toolbox in this work [51]. For training with our PBO, we predefine six candidate λ values $\{5e-2, 1e-2, 5e-3, 1e-3, 5e-4, 1e-4\}$. Unless noted, we train with the cosine annealing learning rate scheduler that gradually decays learning rates from 0.01 to 0 [52].

A. Ablation Study

Tuning Weight Decay Notably Improves Performance. Weight decay is adopted in existing methods [10], [14] (also refer to some released code [40], [41]). However, the literature of CIL underexplores weight decay. Because weight decay encourages learning small weights to avoid overfitting, it tends to “balance” weights by squeezing all weights during training. In this study, when training the classification network (using the cross-entropy loss), we hand-tune a single weight decay hyperparameter λ by setting it to some different values. With a specific λ , we train each model for 320 epochs. Figure 2 plots accuracy curves on the training and validation sets during training with different hyperparameter values. Figure 3 plots the classification accuracy on the validation set as a function of hyperparameter λ . Clearly, manually tuning weight decay boosts performance from 38.6% to 46.5%!

Learning with constant vs. bilevel optimized weight decay. We compare the performance of learning with bilevel optimized weight decay versus a constant weight decay. In this study, we use our PBO algorithm to optimize a single weight decay (rather than per-layer weight decay). Table I compares the results. We can see that learning with bilevel

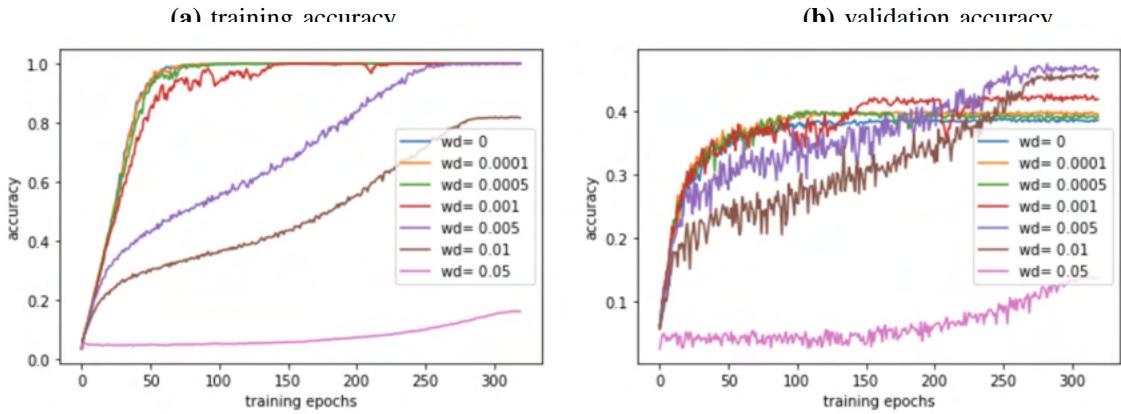


Fig. 2: Accuracy on the training (a) and validation (b) sets by learning with different hyperparameter λ on weight decay terms. When weight decay is too large (e.g., $\lambda=5\text{e-}2$), the model cannot learn well because the weight decay term dominates the overall loss. On the other hand, when weight decay is too small (e.g., as small as zero), the model overfits to the training set quickly and has poor generalization to the validation set. The sweet spot of λ is in the middle, e.g., $\lambda=5\text{e-}3$. We complement this plot with Figure 3 that plots validation accuracy as a function of weight decay.

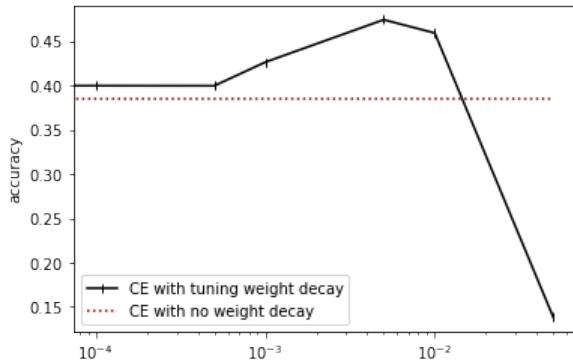


Fig. 3: Tuning weight decay notably improves classification accuracy on the validation set. Specifically, on this benchmark, we find that setting weight decay hyperparameter $\lambda=5\text{e-}3$ yields the best performance, 46.5%, while learning without weight decay achieves 38.6%. Note that tuning weight decay already rivals existing CIL methods, e.g., 38.32% at $\lambda=2\text{e-}4$ [42], [43] (cf. code [40], [41]) and 44.10% at $\lambda=1\text{e-}3$ [23] (cf. code [53]).

optimized weight decay matches the previous best-performing model that is with manually tuned weight decay ($\lambda=5\text{e-}3$). However, our PBO algorithm allows learning with per-layer weight decay, which is expected to perform even better, as studied next.

Cyclic learning rate vs. cosine annealing learning rate. We study the effect of using different learning rate schedules on our PBO algorithm: cosine annealing learning rate [52] and cyclic learning rate [47]. Table II lists the results. We can see that the two different learning rate schedules do not make a difference when learning with constant weight decay, but they do matter in our PBO that bilevel optimizes weight decays which is also shown in Figure 4 (cyclic learning rate yields better performance than cosine annealing learning rate). We conjecture that the cyclic learning rate help training

TABLE I: CIL performance on the validation set by learning with PBO of weight decay versus a single constant weight decay. Here we use PBO to optimize a single weight decay hyperparameter λ shared by all layers. For learning with constant weight decay, we set $\lambda=5\text{e-}3$ which is the best-performing model through manual tuning and validation. PBO matches the the best performance achieved before; note that PBO allows learning with per-layer weight decay, as studied later.

constant WD	PBO WD
46.56 ± 0.16	46.73 ± 0.40

escape current local minima and lands on some better local minimas in the whole course of training.

B. Benchmarking Results

We benchmark our method by comparing against existing CIL methods on this dataset. Table III lists detailed comparison. Clearly, CE, which learns without weight decay, only achieves 38.50% accuracy; when hand-tuning weight decay ($\lambda=5\text{e-}3$) we obtain much better performance 45.75%. In contrast, training with our PBO achieves further improvement, 47.35%. Recall that our PBO method is orthogonal to any existing CIL methods, e.g., τ -normalization which post-hoc normalizes classifiers after training. The τ -normalization notably improves the typical method, e.g., 47.73% by τ -norm on top of a CE model compared to 38.50% by CE. We can also apply the simple $\tau\text{-}\alpha$ -normalization technique to the classifiers learned with our PBO. This achieves the record-breaking result 51.32%!

V. CONCLUSION AND FUTURE WORK

In Class-imbalanced learning (CIL), a naively trained network has imbalanced class-specific weights norms. This motivates us to address CIL via parameter regularization. To

TABLE II: Learning rate scheduler is crucial in our PBO algorithm. We compare classification accuracy (%) on the validation set. When we train with a constant weight decay $\lambda=5e-3$, it does not matter whether to use cyclic learning rate or cosine annealing learning rate. However, it matters in PBO: using cyclic learning rate consistently performs better than cosine annealing learning rate. Intuitively, PBO requires reinitialization and continually training the models, cyclic learning rate helps escape current local minimas and land on better local minimas. In this study, we use PBO on the top one/two layers (weight decay at other layers is set to 5e-3) because we find it does not improve further when used on the rest of layers, presumably because top layers are more class-specific that needs sophisticated regularization.

Method	Cyclic-LR	Cosine-LR
constant WD ($\lambda=0$)	38.45	38.50
constant WD ($\lambda=5e-3$)	46.25	46.35
PBO with single λ	46.21	41.25
PBO on the last layer	46.79	46.52
PBO on the last two layers	47.35	46.17

TABLE III: Top-1 accuracy (%) on the validation set of CIFAR100-LT (where the imbalance factor is 100). Compared with a naively trained model using cross entropy loss (CE) that achieves 38.50%, tuning weight decay (WD) notably improves accuracy (i.e., CE+WD), achieving 45.75%. Learning with PBO algorithm boosts accuracy to 47.35%. Because our PBO algorithm is an orthogonal method to existing ones, we further adopt the τ -normalization atop of the classifiers learned with PBO, yielding the best performance 51.32%.

Method	Validation Accuracy
CE	38.50
CE+CB [10]	39.60
KD [54]	40.36
LDAM-DRW [5]	42.04
BBN [22]	42.56
LogitAjust [55]	42.01
LDAM+SSP [16]	43.43
Focal [56]	38.41
Focal+CB [10]	39.60
De-confound-TDE [23]	44.10
CE with τ -norm [6]	47.73
CE+WD ($\lambda = 5e-3$)	45.75
PBO	47.35
PBO with τ -norm	51.32

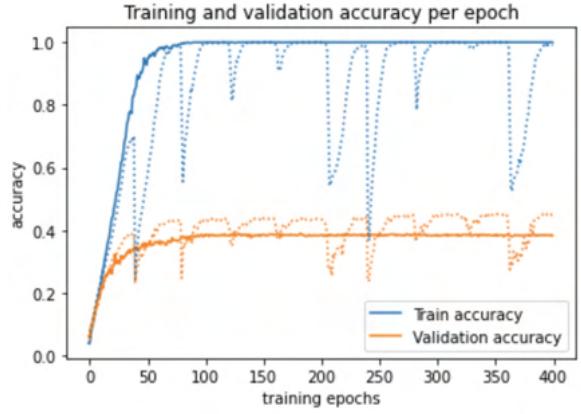


Fig. 4: In our PBO algorithm, using a cyclic learning rate (dotted curves) performs better than cosine annealing learning rate. In each cycle (which consists of 40 epochs), the model achieves a local minima and escape it in the next cycle which yields to a better local minimas eventually.

do so, we propose the population-based bilevel optimization method to optimize per-layer weight decays during training. Our experiments show that learning with bilevel-optimized weight decay achieves the state-of-the-art on the CIFAR100 long-tailed benchmark.

Future Work. While weight decay squeezes network weights towards small values, the (per-class) filter weights are still imbalanced. We are motivated to explore regularization on per-filter weights. We expect our PBO algorithm on weight decay to be also applicable to tuning per-filter weights for better CIL.

ACKNOWLEDGEMENTS

Shaden is supported in part by the KAUST Gifted Student’s Program (KGSP). This work is supported by the CMU Argo AI Center for Autonomous Vehicle Research. The authors would like to thank Rachel Burcin and Dr. John Dolan for their continuous support and organization of this program.

REFERENCES

- [1] W. J. Reed, “The pareto, zipf and other power laws,” *Economics Letters*, vol. 74, no. 1, pp. 15–19, 2001.
- [2] M. Buda, A. Maki, and M. A. Mazurowski, “A systematic study of the class imbalance problem in convolutional neural networks,” 2017.
- [3] B. Z. J. W. Yongshun Zhang, Xiu-Shen Wei, “Bag of tricks for long-tailed visual recognition with deep convolutional neural networks,” 2021.
- [4] X. Yuan, L. Xie, and M. Abouelenien, “A regularized ensemble framework of deep learning for cancer detection from multi-class, imbalanced training data,” *Pattern Recognition*, vol. 77, pp. 160–172, 2018.
- [5] K. Cao, C. Wei, A. Gaidon, N. Aréchiga, and T. Ma, “Learning imbalanced datasets with label-distribution-aware margin loss,” *CorR*, 2019.
- [6] B. Kang, S. Xie, M. Rohrbach, Z. Yan, A. Gordo, J. Feng, and Y. Kalantidis, “Decoupling representation and classifier for long-tailed recognition,” 2019.
- [7] S. H. Khan, M. Hayat, M. Bennamoun, F. Sohel, and R. Togneri, “Cost sensitive learning of deep feature representations from imbalanced data,” 2015.
- [8] E. A. G. Haibo He, “Learning from imbalanced data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 9, no. 21, 2009.

- [9] S. Zhang, C. Chen, X. Hu, and S. Peng, “Balanced knowledge distillation for long-tailed learning,” 2021.
- [10] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, “Class-balanced loss based on effective number of samples,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.
- [11] M. A. Jamal, M. Brown, M.-H. Yang, L. Wang, and B. Gong, “Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective,” 2020.
- [12] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” 2011.
- [13] F. Alberto, G. Salvador, M. Galar, R. C. Prati, B. Krawczyk, and F. Herrera, *Learning from imbalanced data sets*. Springer Science+Business Media, 2018.
- [14] B. Kang, S. Xie, M. Rohrbach, Z. Yan, A. Gordo, J. Feng, and Y. Kalantidis, “Decoupling representation and classifier for long-tailed recognition,” in *International Conference on Learning Representations*, 2020.
- [15] Y.-X. Wang, D. Ramanan, and M. Hebert, “Learning to model the tail,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.
- [16] Y. Yang and Z. Xu, “Rethinking the value of labels for improving class-imbalanced learning,” 2020.
- [17] C. Drummond, R. C. Holte, et al., “C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling,” in *Workshop on learning from imbalanced datasets II*, vol. 11. Citeseer, 2003, pp. 1–8.
- [18] H. Han, W.-Y. Wang, and B.-H. Mao, “Borderline-smote: A new over-sampling method in imbalanced data sets learning,” in *Advances in Intelligent Computing*, D.-S. Huang, X.-P. Zhang, and G.-B. Huang, Eds., 2005.
- [19] S. H. Khan, M. Hayat, W. Zamir, J. Shen, and L. Shao, “Striking the right balance with uncertainty,” *CoRR*, 2019.
- [20] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker, “Feature transfer learning for face recognition with under-represented data,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.
- [21] S. Liu, R. Garrepalli, T. G. Dietterich, A. Fern, and D. Hendrycks, “Open category detection with pac guarantees,” in *ICML*, 2018.
- [22] B. Zhou, Q. Cui, X.-S. Wei, and Z.-M. Chen, “Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.
- [23] K. Tang, J. Huang, and H. Zhang, “Long-tailed classification by keeping the good and removing the bad momentum causal effect,” in *Advances in Neural Information Processing Systems*, 2020.
- [24] P. Bühlmann and S. Van De Geer, *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.
- [25] I. Goodfellow, Y. Bengio, and A. Courville, “Regularization for deep learning,” *Deep learning*, pp. 216–261, 2016.
- [26] J. Kukavcka, V. Golkov, and D. Cremers, “Regularization for deep learning: A taxonomy,” *arXiv preprint arXiv:1710.10686*, 2017.
- [27] T. Salimans and D. P. Kingma, “Weight normalization: A simple reparameterization to accelerate training of deep neural networks,” 2016.
- [28] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” *arXiv preprint arXiv:1607.06450*, 2019.
- [29] L. Prechelt, “Early stopping-but when?” in *Neural Networks: Tricks of the trade*. Springer, 1998, pp. 55–69.
- [30] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, “mixup: Beyond empirical risk minimization,” in *International Conference on Learning Representations*, 2018.
- [31] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arXiv preprint arXiv:1207.0580*, 2012.
- [32] A. Y. Ng, “Feature selection, l 1 vs. l 2 regularization, and rotational invariance,” in *International Conference on Machine Learning*, 2004.
- [33] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.
- [34] L. N. Smith, “A disciplined approach to neural network hyperparameters: Part 1—learning rate, batch size, momentum, and weight decay,” *arXiv preprint arXiv:1803.09820*, 2018.
- [35] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization.” *Journal of machine learning research*, vol. 13, no. 2, 2012.
- [36] D. Ha, A. M. Dai, and Q. V. Le, “Hypernetworks,” *CoRR*, 2016.
- [37] A. Sinha, T. Khandait, and R. Mohanty, “A gradient-based bilevel optimization approach for tuning hyperparameters in machine learning,” *CoRR*, 2020.
- [38] M. MacKay, P. Vicol, J. Lorraine, D. Duvenaud, and R. B. Grosse, “Self-tuning networks: Bilevel optimization of hyperparameters using structured best-response functions,” *CoRR*, 2019.
- [39] L. Li, K. G. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Efficient hyperparameter optimization and infinitely many armed bandits,” *CoRR*, 2016.
- [40] K. Cao, “<https://github.com/kaidic/LDAM-DRW/blob/3193f05c1e6e8c4798c5419e97c5a479d991e3e9/cifar.train.py#L54>,” commit 6feb304, 2019.
- [41] Y. Cui, “https://github.com/richardaecn/class-balanced-loss/blob/1d7857208a2abc03d84e35a9d5383af8225d4b4d/src/cifar_main.py#L581,” commit 0ab6eb7, 2019.
- [42] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, “Class-balanced loss based on effective number of samples,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.
- [43] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, “Learning imbalanced datasets with label-distribution-aware margin loss,” in *Advances in Neural Information Processing Systems*, 2019.
- [44] K. Jamieson and A. Talwalkar, “Non-stochastic best arm identification and hyperparameter optimization,” in *Artificial Intelligence and Statistics*. PMLR, 2016, pp. 240–248.
- [45] L. Li, K. G. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, “Hyperband: Bandit-based configuration evaluation for hyperparameter optimization,” in *ICLR*, 2017.
- [46] M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan, et al., “Population based training of neural networks,” *arXiv preprint arXiv:1711.09846*, 2017.
- [47] L. N. Smith, “Cyclical learning rates for training neural networks,” in *2017 IEEE winter conference on applications of computer vision (WACV)*. IEEE, 2017, pp. 464–472.
- [48] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, “Snapshot ensembles: Train 1, get m for free,” *arXiv preprint arXiv:1704.00109*, 2017.
- [49] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, “Large-scale long-tailed recognition in an open world,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.
- [50] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *CVPR*, 2016.
- [51] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al., “Pytorch: An imperative style, high-performance deep learning library,” *Advances in neural information processing systems*, vol. 32, pp. 8026–8037, 2019.
- [52] I. Loshchilov and F. Hutter, “Sgdr: Stochastic gradient descent with warm restarts,” 2017.
- [53] K. Tang, “https://github.com/KaihuaTang/Long-Tailed-Recognition_pytorch/blob/90c8b2c0b66d17f78b67263861bc9d858fe20128/classification/config/CIFAR100_LT/feat_unifrom.yaml#L22,” commit 54c07cf, 2020.
- [54] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” 2015.
- [55] A. K. Menon, S. Jayasumana, A. S. Rawat, H. Jain, A. Veit, and S. Kumar, “Long-tail learning via logit adjustment,” in *International Conference on Learning Representations*, 2021.
- [56] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” in *Proceedings of the IEEE international conference on computer vision*, 2017.

An Affordable and Accessible Educational Manipulator

Samuel C. Alvares¹, Pragna Mannam², Oliver Kroemer², Zeynep Temel²

Abstract—Educational robots provide an engaging platform to teach students interdisciplinary skills and inspire them to pursue careers in STEM. Currently, most educational robots are wheeled. These platforms can be effective at introducing engineering and coding concepts to a wide range of students when provided at a low cost and with educational resources. An alternative perspective to developing an educational robot is a manipulator, or a stationary robot that can interact with its environment. Traditionally used in factories, manipulators were expensive and unsafe for an educational setting. Current technologies allow us to 3D-print manipulators from soft materials so that they are safe for student use and low cost. To progress towards a safe, affordable, and accessible manipulator for educating students 12 to 18 years of age, we present DeltaZ. Unlike a traditional wheeled robot, DeltaZ sits on a desktop and introduces students to the ideas of kinematics, sensing, and manipulation. This delta-style robot has three translational degrees of freedom and closed-form kinematic solutions which make manipulation problems more intuitive compared to other manipulators. Clear instructional resources are provided with various learning stages to engage students from a novice to advanced roboticist. Open-source 3D-printable designs and code are made available to the public to modify, improve and share back to the community. Open-hardware designs, open-source code, and educational materials are available at <https://github.com/alvaresc/DeltaZ>.

Index Terms—Education Robotics, Parallel Robots, Compliant Joints and Mechanisms, Additive Manufacturing, Soft Robot Applications, Flexible Robotics, Kinematics

I. INTRODUCTION

Robot manipulators consist of a series of bodies interconnected by articulated joints and have a means for performing a task [1]. Not only are manipulators effective at replacing humans in dangerous or repetitive tasks, they are becoming increasingly important in minimally invasive surgery [2], working in space [3], and even collaborating with humans [4]. It is important to inspire today's youth to engage in STEM learning and continue development in fields such as manipulation. Educational robots are becoming more prevalent in the classroom [5] and are an effective means to engage students in learning STEM subjects such as robotics, computer science, electrical engineering, mechanical design, and mechatronics. Most educational robots are wheeled [6]–[9]. These robots can explore their environment via remote control or autonomously with sensors, and some even have trainings and certificates for teachers to better adopt these

¹S.C. Alvares is with the Robotics Institute Summer Scholars at Carnegie Mellon University, Pittsburgh, PA 15213, USA and also with Rose-Hulman Institute of Technology, Terre Haute, IN 47803, USA alvaresc@rose-hulman.edu

²P. Mannam, O. Kroemer, and Z. Temel are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA {pmannam, okroemer, ztemel} @andrew.cmu.edu



Fig. 1: The mechanical design of DeltaZ. The upper part of the robot encloses all motors and electronics and is supported by a tripod base. The 3D-printed soft delta component (white) has flexible joints that enable the robot to move similarly to a rigid robot.

robots in the classroom [10]. With clear educational goals and unifying concepts, educational robots can be an effective tool to teach STEM principles and give students the confidence to pursue STEM-related careers. There are, however, few affordable educational robots that introduce students to the ideas of manipulation. This is because most manipulators are too dangerous, expensive, or complicated to be viable for teaching middle or high school students. Recent technologies have allowed us to fabricate low-cost manipulators using flexible, 3D-printed materials [11]. Based on this work, DeltaZ, shown in Fig. 1, presents a key step towards an affordable, accessible, manipulator-style educational robot geared towards students 12 to 18 years of age. The robot uses key physical parameters from [11] but offers an improved mechanical design and is tailored to a learning environment. Its design features a modernistic, 3D-printed enclosure for electronics and safe, 3D-printed, flexible joints. Due to its three translational degrees of freedom, DeltaZ provides a straightforward platform to introduce students to the fundamental ideas associated with manipulators including setup, kinematics, manipulation, and sensing. This robot also has open-source software and hardware which encourages communities of users to share ideas and collaborate on improving the overall design and functionality of the robot [12].

II. RELATED WORKS

An engaging way to teach students about STEM-related topics is through educational robots; previous work has demonstrated the effectiveness of such robots in education. One study on 23 seventh grade students showed that robotics activities led to an increased interest and higher self-efficacy in STEM tasks [13]. Thus, being exposed to robotics in high school or college may give students the confidence to pursue STEM-related classes and careers. Another study investigated the impacts of educational robotics on technical and social skills [14], and it showed that students who worked with robotics in middle and high school not only improved their technical skills in areas such as math and programming, but users had more positive views on STEM fields. In addition, students who worked with robotics in their education demonstrated improved soft skills such as teamwork. Other work is being done to investigate additional benefits of robots in classrooms such as boosting female involvement in STEM [15]. Certain educational robot manufacturers, such as Thymio, are placing a special emphasis on creating gender-neutral designs to boost the inclusivity of their products [7]. Thymio also boasts low costs and multiple coding environments to appeal to a wide range of users.

There are a number of successful, low-cost, open-source educational robots. Hapkit is an example of an educational robot that promotes open-hardware design and even provides training on how to customize their hardware in a free CAD software [16]. Other open-source educational robots include, among others, the Edison [9], Finch [8], and Thymio [7]. These robots, like most educational robots, are wheeled and teach robotics concepts such as coding, sensing, and autonomous navigation. Manipulators, although common in industry, are under-represented in educational robots. Elenco manufactures two low-cost educational robot manipulator kits. The HydroBot [17] is a hydraulic robot manipulator, and the Wire Controlled Robot Arm [18] is a serial manipulator controlled with motors. These robots are marketed to teach STEM though hands-on activities. There are, however, few other low-cost educational manipulators available. The European Lab for Educational Technology provides a more thorough review of available educational robots [19].

Certain challenges exist with incorporating robots into educational programs. In particular, many robots lack the educational materials necessary to prepare both students and teachers to adopt a robot in a class's curriculum [19]. VEX robotics is tackling this problem by hosting in-person teacher training courses and certificates in robotics [10]. VEX educational robots do, however, have a higher price point compared with other competitors such as Thymio, Finch, and Edison, which limits accessibility. [20] points out key actions to increase robot usage in the classroom, including the need for educational robot workshops at one of the main robotic conferences as well the need for government implementation of widespread robotics curricula.

Soft, delta-style robots have excellent potential for use as an educational tool due to their high performance, low-cost,

TABLE I: Stakeholders and their perspectives in terms of an educational manipulator robot.

Stakeholder	Perspective
Middle and high school students	Want to be engaged and challenged, prefer a straightforward fabrication, setup, and assembly process
Middle and high school teachers	Want to inspire students and teach with technology, value low cost and clear educational materials
Independent STEM organizations	Want to increase student involvement in STEM and inspire future scientists, want a durable robot
The present lab	Wants to develop a platform that helps students engage in STEM, placing an emphasis on affordability and accessibility

inherent safety, and intuitive motions. Delta-style robots have base mounted motors and parallel geometry that allow for fast and accurate motions with relatively small and low-cost motors [21]–[23]. Previous work has been done to develop a 3D-printable soft delta robot design which mimics the kinematics of the rigid counterpart [11], [24]. TPU (thermoplastic polyurethane) 95A, which is used to 3D-print low-cost soft delta robots in [11], [24], is compliant and safe for student use. In addition, a closed-form kinematic solution and three translational degrees of freedom make the robot's motions intuitive compared to other manipulators such as the previously mentioned Elenco robot arms [17], [18].

DeltaZ builds upon the previous work of [11] in which a 3D-printed, soft, delta-style robot was initially designed and evaluated using revolute actuators. Using key physical parameters from the original work, we improved upon the mechanical design of a soft, delta-style robot by making it more conducive for a learning environment. In particular, mechanical connections were made more durable and reliable for educational use. Also, an encasement was designed to house all electronics and motors. A universal end-effector attachment allows a variety of manipulation tasks to be achieved, making the robot more engaging for students. In addition to hardware contributions, software resources include a user-friendly GUI, educational materials and videos, and an Arduino library to handle forward and inverse kinematics based upon related work [23]. The design of DeltaZ's hardware and software are tailored for educational use.

III. METHODOLOGY

The design of DeltaZ's software and hardware was derived from stakeholder needs and perspectives. Since DeltaZ is primarily be used as an educational tool, main stakeholders include middle and high school students and teachers. Other stakeholders include independent STEM organizations and the present research lab. Members of all stakeholder groups were interviewed with a common set of questions. A summary of stakeholder perspectives is shown in Table I.

From the stakeholder perspectives, we identified 6 primary needs. In order of importance, DeltaZ needs to be:

- 1) affordable
- 2) accessible
- 3) engaging
- 4) educational
- 5) easy to fabricate, setup and assemble
- 6) durable.

Low cost is critical to making DeltaZ available to a broad range of users. Other successful low cost manipulators such as Thymio, Edison and Finch cost \$189, \$49, and \$139, respectively, so we establish low cost in this context at around \$100 to \$200 [7]–[9]. We target DeltaZ to have a cost of under \$100 which will allow a classroom set of 10 robots to be purchased for under \$1000.

A focus on accessibility, in terms of obtaining, using, and learning, is important to make the robot as inclusive as possible. We target having open-source software and hardware which allows anyone to use and modify existing hardware or software for free. Designs should be approachable and inclusive for users. For the mechanical design, we target gender-neutrality which can be achieved through color and a minimalist design with a mix of curved and sharp edges. Also, user interfaces should be made approachable with color and intuitive controls. Presenting educational materials in both written and video format makes learning more accessible to those with different leaning styles. Also, video subtitles make educational videos accessible to a wider range of users.

Engagement is important to encourage students to operate and learn using DeltaZ. Since students 12 to 18 years of age have a large range of experience levels, DeltaZ should be educational for students who are early beginners to those who are experienced roboticists. One way to achieve this is to have various learning stages of increasing difficulty. In this way, the barrier to entry is low for beginners, and advanced students will be challenged in the final stages. A user experience study could provide feedback to quantify how engaging the robot really is. We are targeting a large majority of over 75% positivity on a simple feedback survey which asks users to rate the positivity of their experience with DeltaZ on a scale from 1 to 10.

DeltaZ is an educational robot and needs to convey fundamental topics in manipulation such as setup, assembly, forward and inverse kinematics, manipulation of the environment, and sensing. Resources such as educational videos to accompany written materials, references for further learning, as well as options for creative learning can increase the educational value of the robot. Videos, which address visual and auditory learning styles, enhance the instructive abilities of the robot. Also, hands on demonstrations help convey the main learning objectives. References for further learning will allow students to expand their knowledge in areas of interest. Multiple stakeholders mentioned the need for "creative learning" in which users would have the opportunity to dive deeper into materials by devising their own unique solutions to open-ended design problems.

A positive user experience is promoted by simple and straightforward assembly, fabrication, and setup processes.

TABLE II: DeltaZ needs mapped to target specifications

Need	Target
1) Affordable	a) total cost under \$100
2) Accessible	a) open-source software/hardware b) gender-neutral design
3) Engaging	a) >75% positivity on survey b) multiple learning stages
4) Educational	a) educational videos provided b) references for further learning provided c) options for creative learning
5) Easy to fabricate, setup and assemble	a) <50 components b) build time <50 minutes
6) Durable	a) >5000 cycles in endurance test

A simple design with clear instructions can benefit a user's experience during these steps. A typical high school or middle school class is at least 50 minutes long. To allow the robot to be built in one class period, 50 minutes is set as the upper target on mechanical build time. Assuming that each component takes about 1 minute to install, we are targeting to have a design with under 50 total components. Also, 3D-printing allows custom parts to be fabricated easily and affordably.

Durability is given the lowest importance because the robot will be primarily 3D-printed, so the user can reprint damaged or worn components. However, it is important that the robot can withstand the stresses of a learning environment without frequent repair or maintenance. An endurance test is used to quantify the durability of the robot. One endurance cycle is completed when the robot travels to the top and bottom of its workspace as well as around the circumference of its workspace. Roughly, we expect the robot to go through 50 cycles each time the robot is used. If an individual were learning with DeltaZ, he or she may use the robot 5 times per week over the course of a month. If the robot were to last 5 years, then the robot would need to endure at least

$$\frac{50 \text{ cycles}}{\text{use}} \cdot \frac{5 \text{ uses}}{\text{week}} \cdot \frac{4 \text{ weeks}}{\text{month}} \cdot 5 \text{ years} = 5000 \text{ cycles}.$$

Target metrics, which are established to objectively assess the performance of the robot's design, are summarized in Table II with the associated robot needs. DeltaZ's designs are driven by the established metrics.

IV. DESIGN

Affordability and accessibility, in addition to other criteria discussed in Section III, are central to the design of DeltaZ's hardware, software, and educational materials.

The overall mechanical design of DeltaZ is shown in Fig. 1. Inclusivity of the mechanical design was achieved through a low-cost, gender-neutral design, and simple fabrication and assembly processes. The base cost of the robot is affordable compared to similar low-cost educational robots, at around \$46. Depending on the user's needs and abilities, various sensors or add-ons can be purchased to expand upon the

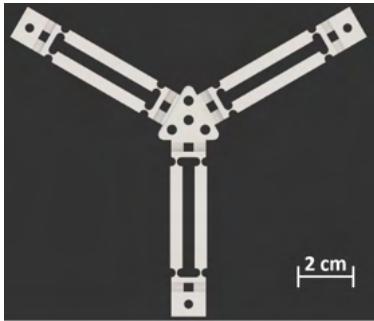


Fig. 2: The soft delta component of DeltaZ can be 3D-printed with 4 central holes to attach a variety of end-effectors using M3 screws. This component bolts to the rigid forearms via the outer 3 holes.

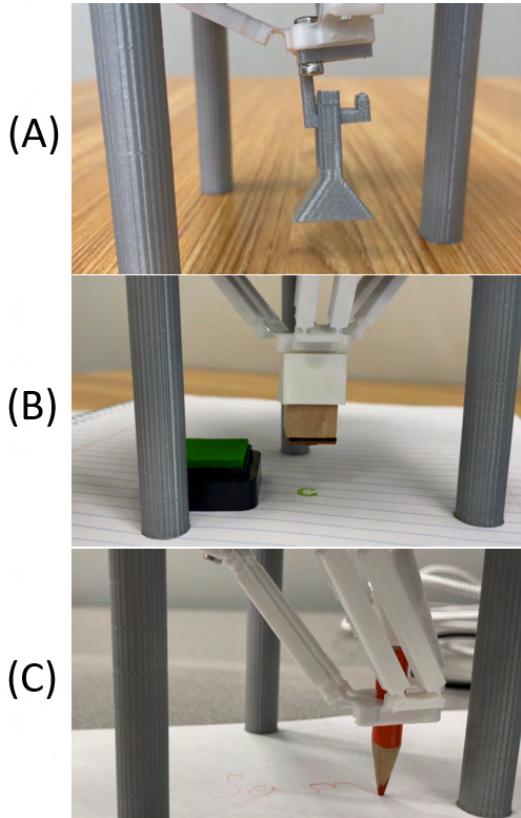


Fig. 3: Examples of how DeltaZ can interact with its environment in a useful or entertaining way including: (A) a hook end-effector picking up a weight, (B) stamping a letter with an ink pad and a rubber stamp, and (C) drawing with a colored pencil.

base robot. The mechanical design is intentionally gender-neutral as to not preclude user groups. Specifically, DeltaZ is minimalist in its design, and it features a variety of curved and sharp edges to not appeal to one gender in particular. In terms of fabrication, all custom components are 3D-printed. The rigid components of the robot, shown in black in Fig. 1, are 3D-printed from PLA, and the white component in Fig. 1, referred to as the soft delta, is printed

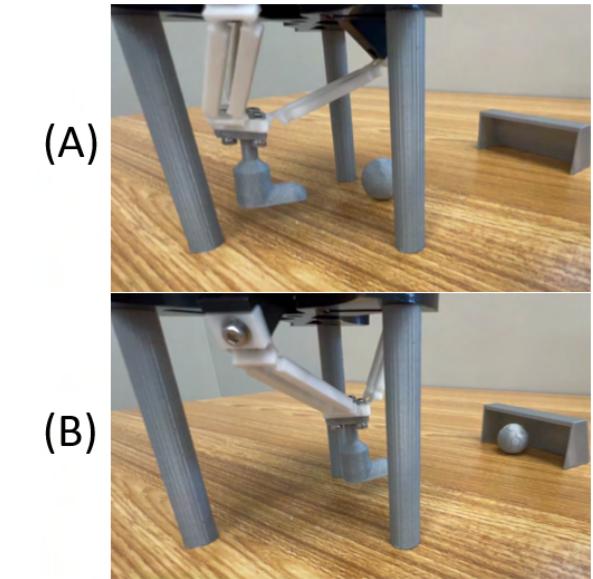


Fig. 4: (A) DeltaZ, with a 3D-printed boot attachment, winds up to kick a soccer ball. (B) DeltaZ kicks the ball into the goal.

from a flexible material, TPU 95A, as in [11]. 3D-printing the soft delta with TPU is affordable and accessible to anyone who has access to a 3D-printer. Even if an individual does not have access to a 3D-printer, 3D-printing services such as CraftCloud [25] will print any or all of the components. The soft delta is 3D-printed as one piece, reducing the number of components and assembly steps. DeltaZ is made of 42 total pieces, including individual screws, all of which are screwed or bolted together. A video on DeltaZ’s website walks users through the assembly process. The time to assemble DeltaZ’s 42 components varies based upon the user’s experience level. More experienced users can assemble the robot in under 30 minutes, but other users may take up to 50 minutes. DeltaZ is durable and is rated for 5 years of regular classroom usage as it passed a 5000 cycle endurance test.

The functionality of DeltaZ is enabled by its three translational degrees of freedom. To leverage this movement and teach the ideas of manipulation, the soft delta design allows various end-effectors to be attached using an array of small bolts, as shown in Fig. 2. Fig. 3(A), Fig. 3(B), and Fig. 4 provide examples of DeltaZ completing manipulation tasks with 3D printed end-effectors. As shown in Fig. 3(C), an alternative design of the soft delta component of DeltaZ allows for a colored pencil to be inserted for drawing. Users are instructed on how to use TinkerCAD, a free online CAD software, to design custom end-effectors for DeltaZ. With this knowledge and some user creativity, there are limitless manipulation tasks that DeltaZ could achieve.

DeltaZ’s software also promotes accessibility, ease of use, and benefits the educational goals of the robot. DeltaZ’s software package includes an Arduino library to handle forward and inverse kinematics, a communications function to handle messages, and a graphical user interface (GUI) to

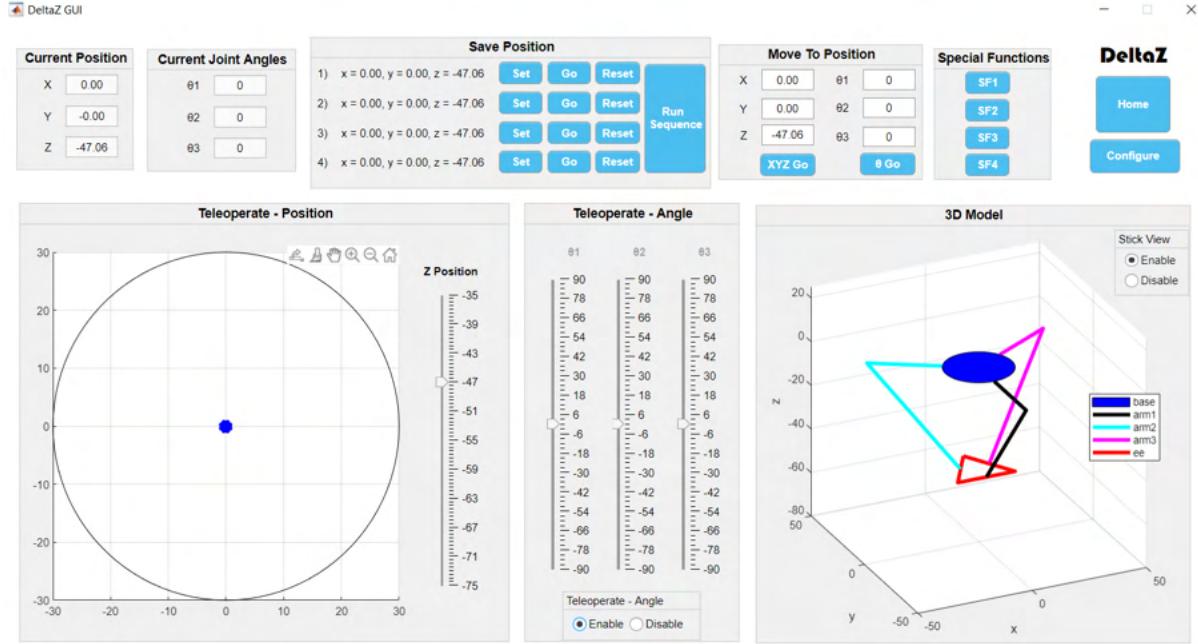


Fig. 5: The DeltaZ GUI which allows users to control the robot using either forward or inverse kinematics and see a stick model of the robot’s pose in real time. Users can also save positions, configure the workspace, or even code their own own special functions in Arduino and run them from the GUI.

send commands to the robot and display its state. DeltaZ’s GUI, shown in Fig. 5, allows movement of DeltaZ without any text coding input from the user. Thus, those who have no coding experience can learn about manipulation with DeltaZ. The Arduino platform allows individuals with more coding experience to develop their own text code in Arduino’s simple and open-source coding environment. The GUI allows users to control the robot by setting x-, y-, and z-positions or by setting the joint angles of the robot. Therefore, it is intuitive to teleoperate the robot and to learn forward and inverse kinematics with DeltaZ. The workspace of the robot can be adjusted to prevent crashes when using various attachments. Other functionalities of the GUI include saving positions and a stick figure which updates pose in real-time with physical robot.

DeltaZ provides written and video-based educational materials with a central theme of manipulation. The learning stages are: materials, building, setup, moving, manipulation, and sensing. In the materials and setup sections, users are given a linked bill of materials and instructions to setup the robot’s software. Users are introduced to forward and inverse kinematics concepts and how to operate the GUI in the moving stage. The manipulation section of DeltaZ walks students through various manipulation examples and provides a short design overview so users can create their own end-effectors. The final lesson of DeltaZ involves sensing and analog inputs. Instructional materials are provided on how to connect various inputs, such as a potentiometer, a force sensitive resistor, and a joystick, to control the position of the robot. The user is tasked with combining multiple inputs to add intelligence to the robot. At least one

TABLE III: DeltaZ needs mapped to target specifications

Target	Evaluation
1a) cost <\$100	Met: base cost is \$46
2a) open-source 2b) gender neutral	Met: open-source software/hardware Met: gender neutral design
3a) positivity score 3b) multiple stages	Undetermined: see Future Work Met: 6 total stages of increasing difficulty
4a) educational videos 4b) learning references 4c) creative options	Met: 7 educational videos Met: references provided for further learning Met: incorporated creative learning
5a) <50 components 5b) build time	Met: 42 components Met: build time around 30 minutes
6a) endurance test	Met: >5000 endurance cycles completed

video is provided for each learning stage of DeltaZ, and all videos have subtitles to make them more accessible. The sections of DeltaZ increase in difficulty so that beginning and advanced users can gain from their experience with DeltaZ. Also, there are a variety of open ended design challenges and many references provided to contribute to the learning and inspiration of users. The 6 lessons of DeltaZ, videos, and open-source software and hardware are located at <https://github.com/alvaresc/DeltaZ>.

A summary of the performance of DeltaZ’s software and hardware designs in terms of established target metrics is shown in Table III. Although initial user experiences have been positive, more in depth user experience studies needs to be conducted.

V. DISCUSSION AND CONCLUSION

DeltaZ is an educational robot that is affordable, accessible, and engaging for middle- and high school-aged students. It addresses science, technology, engineering, and math in an exciting and engaging manner, and it aims to introduce students to key ideas in manipulation. DeltaZ is for a broad range of students from someone who has never interacted with a robot to a fourth-year high school robotics student. After identifying and interviewing stakeholder groups, we ranked needs for the robot and determined metrics to assess the effectiveness of the design. DeltaZ's low cost, free software and designs, and simple instructions reduce the barrier to entry. Various learning stages teach principles such as forward and inverse kinematics, manipulation strategies, and sensing. Distinct learning stages with written and filmed materials make DeltaZ engaging and educational. The joints of the robot are printed from a flexible material, TPU, making the design affordable and accessible to anyone who has access to a 3D-printer, or a 3D-printing service. Overall, user experiences with the robot have been positive. DeltaZ is a robot that has the potential to inspire future scientists and engineers, and it provides a unique platform for an educational robot which is affordable and accessible.

VI. FUTURE WORK

Among the key aspects for the future development for DeltaZ are to conduct more in-depth user experience surveys, to implement closed-loop position control by sensorizing the robot, and to refine the educational resources. The DeltaZ platform allows users to expand upon the robot in any way that they find interesting including modifying existing designs, designing new components and end effectors, programming new functionalities, and then to share this work back to the community. Future revisions of DeltaZ will take these contributions into consideration. We hope to continue increasing the accessibility of DeltaZ by implementing features to improve user experience for people with disabilities.

ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under Grant No. 2024794. I would like to thank RISS for welcoming me back to the program for the second time. Thank you to Rachel Burcin, John Dolan, and Jennie Piotrkowski for your diligent efforts maintaining the RISS community while still online. Next, thank you to my mentor, Pragna Mannam, for her commitment to helping me succeed in my project. I would also like to thank Dr. Zeynep Temel, and Dr. Oliver Kroemer for supporting my project even with busy schedules. Pragna Mannam, Dr. Zeynep Temel, and Dr. Oliver Kroemer take full credit for the initial design of the 3D-printable soft delta that was instrumental in this work. Thank you to all the individuals who I interviewed for this design and your thoughtful input and time. Finally, I am thankful for the support of the other members of RISS.

REFERENCES

- [1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer Science & Business Media, Aug. 2010.
- [2] E. Vander Poorten, C. N. Riviere, J. J. Abbott, C. Bergeles, M. A. Nasser, J. U. Kang, R. Sznitman, K. Faridpooya, and I. Iordachita, "Robotic Retinal Surgery," in *Handbook of Robotic and Image-Guided Surgery*. Elsevier, 2020, pp. 627–672. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/B9780128142455000360>
- [3] Y. Xu, H. Brown, M. Friedman, and T. Kanade, "Control system of the self-mobile space manipulator," *IEEE Transactions on Control Systems Technology*, vol. 2, no. 3, pp. 207–219, Sept. 1994, conference Name: IEEE Transactions on Control Systems Technology.
- [4] R. Liu and C. Liu, "Human Motion Prediction Using Adaptable Recurrent Neural Networks and Inverse Kinematics," *IEEE Control Systems Letters*, vol. 5, no. 5, pp. 1651–1656, Nov. 2021, conference Name: IEEE Control Systems Letters.
- [5] F. B. V. Benitti, "Exploring the educational potential of robotics in schools: A systematic review," *Computers & Education*, vol. 58, no. 3, pp. 978–988, Apr. 2012. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0360131511002508>
- [6] F. Arvin, J. Espinosa, B. Bird, A. West, S. Watson, and B. Lennox, "Mona: an Affordable Open-Source Mobile Robot for Education and Research," *Journal of Intelligent & Robotic Systems*, vol. 94, no. 3, pp. 761–775, June 2019. [Online]. Available: <https://doi.org/10.1007/s10846-018-0866-9>
- [7] F. Mondada, M. Bonani, F. Riedo, M. Brio, L. Pereyre, P. Retornaz, and S. Magnenat, "Bringing Robotics to Formal Education: The Thymio Open-Source Hardware Robot," *IEEE Robotics Automation Magazine*, vol. 24, no. 1, pp. 77–85, Mar. 2017, conference Name: IEEE Robotics Automation Magazine.
- [8] T. Lauwers and I. Nourbakhsh, "Designing the Finch: Creating a Robot Aligned to Computer Science Concepts," in *First AAAI Symposium on Educational Advances in Artificial Intelligence*, July 2010. [Online]. Available: <https://www.aaai.org/ocs/index.php/EAAI/EAAI10/paper/view/1849>
- [9] "Edison Programmable Robot - Ideal for school classroom education." [Online]. Available: <https://meetedison.com/>
- [10] "VEX Robotics." [Online]. Available: <https://www.vexrobotics.com/>
- [11] P. Mannam, O. Kroemer, and F. Z. Temel, "Characterization of Compliant Parallelogram Links for 3D-Printed Delta Manipulators," in *Experimental Robotics*, B. Siciliano, C. Laschi, and O. Khatib, Eds. Cham: Springer International Publishing, 2021, vol. 19, pp. 75–84, series Title: Springer Proceedings in Advanced Robotics. [Online]. Available: https://link.springer.com/10.1007/978-3-030-71151-1_7
- [12] "Open Source Hardware Association -." [Online]. Available: <https://www.oshwa.org/>
- [13] T. B. Hinton, M. L. Rice, A. Benson, R. Mayben, R. Odom-Bartel, and V. Wright, "AN EXPLORATORY STUDY OF A ROBOTICS EDUCATIONAL PLATFORM ON STEM CAREER INTERESTS IN MIDDLE SCHOOL STUDENTS," p. 158.
- [14] M. Kandlhofer and G. Steinbauer, "Evaluating the impact of educational robotics on pupils' technical- and social-skills and science related attitudes," *Robotics and Autonomous Systems*, vol. 75, pp. 679–685, Jan. 2016. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0921889015001955>
- [15] G. Venture, "Can Robots in Classrooms Attract More Women to Engineering? [Women in Engineering]," *IEEE Robotics Automation Magazine*, vol. 21, no. 4, pp. 130–131, Dec. 2014, conference Name: IEEE Robotics Automation Magazine.
- [16] M. O. Martinez, J. Campion, T. Gholami, M. K. Rittkaidachar, A. C. Barron, and A. M. Okamura, "Open source, modular, customizable, 3-D printed kinesthetic haptic devices," in *2017 IEEE World Haptics Conference (WHC)*, June 2017, pp. 142–147.
- [17] "HydroBot Arm Kit." [Online]. Available: <https://shop.elenco.com/consumers/hydrobot-arm-kit.html>
- [18] "Robotic Arm Wire Controlled." [Online]. Available: <https://shop.elenco.com/consumers/robotic-arm-wire-controlled.html>
- [19] T. Sapounidis and D. Alimisis, "Educational robotics for STEM: A review of technologies and some educational considerations," Sept. 2020, pp. 167–190.
- [20] A. Bredenfeld, A. Hofmann, and G. Steinbauer-Wagner, "Robotics in Education Initiatives in Europe-Status, Shortcomings and Open Questions," Jan. 2010.

- [21] M. López, E. Castillo, G. García, and A. Bashir, “Delta robot: Inverse, direct, and intermediate Jacobians,” *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 220, no. 1, pp. 103–109, Jan. 2006. [Online]. Available: <http://journals.sagepub.com/doi/10.1243/09544060X78263>
- [22] H. McClintock, F. Z. Temel, N. Doshi, J.-s. Koh, and R. J. Wood, “The milliDelta: A high-bandwidth, high-precision, millimeter-scale Delta robot,” *Science Robotics*, vol. 3, no. 14, Jan. 2018, publisher: Science Robotics Section: Research Article. [Online]. Available: <https://robotics.sciencemag.org/content/3/14/eaar3018>
- [23] “Delta robot kinematics,” July 2009. [Online]. Available: <https://hypertriangle.com/ alex/delta-robot-tutorial/>
- [24] P. Mannam*, A. Rudich*, K. Zhang*, M. Veloso, O. Kroemer, and F. Temel, “A Low-Cost Compliant Gripper Using Cooperative Mini-Delta Robots for Dexterous Manipulation,” in *Robotics: Science and Systems XVII*. Robotics: Science and Systems Foundation, July 2021. [Online]. Available: <http://www.roboticsproceedings.org/rss17/p076.pdf>
- [25] “Craftcloud® by All3DP | Your 3D Printing Service Marketplace.” [Online]. Available: <https://craftcloud3d.com/>

Synthetic Data Generation for the Natural Language Component of an Artificial Social Intelligence Agent

Ryan Aponte¹, Aishwarya Jadhav², Joseph Campbell², Dana Hughes², and Katia Sycara²

Abstract—This paper describes a data synthesis method used to improve the feedback of an artificial social intelligence (ASI) agent as part of the Artificial Social Intelligence for Successful Teams (ASIST) project. We use Natural Language Processing (NLP) to understand and predict transcribed speech for the agent. Currently, human teams may make errors due to the chaotic nature of disasters. An artificial social intelligence agent can assist human teams by recognizing and correcting these errors. The agent uses a Theory of Mind (ToM) to model humans in a simulated search-and-rescue environment; it estimates the state of a participant's mind to make predictions about actions. The communication analysis module, a competent of the agent, labels human utterances with a hierarchical coding scheme. These labels help update the agent's ToM models of the humans. The ASIST project collects data from a simulated search-and-rescue environment in Minecraft. The communication analysis module is trained with human subject transcripts from the Minecraft environment which are then coded and prepared to be merged with message bus data. Due to significant limitations in collecting data for the project, data synthesis techniques are discussed.

Index Terms—human performance augmentation, agent-based systems

I. INTRODUCTION

Humans use a Theory of Mind (ToM) to infer the mental states and make predictions about other humans [1]. This is considered a theory due to the state of another individual's mind not being directly observable and because the ToM makes predictions [1]. A simple use of a ToM is to provided as an example.

Maxi eats half his chocolate bar and puts the rest away in the kitchen cupboard. Then he goes out to play in the sun. Meanwhile Maxi's mother comes into the kitchen, opens the cupboard and sees the chocolate bar. She puts it in the fridge. When Maxi comes back into the kitchen, where will he look for his chocolate bar? [3]

Maxi will look for the chocolate bar in the cupboard, since this is where Maxi left it [3].

Recognizing and understanding verbal communication is valuable for a ToM. When verbal communication is used, the ASI agent has an opportunity to update its ToM. For example, person A may tell person B that a victim has been found in room 203. At this point, the ASI agent should

¹Ryan Aponte is with the Herbert Wertheim College of Engineering, University of Florida, Gainesville, FL 32611, USA. aponte.ryan@ufl.edu

²Aishwarya Jadhav, Joseph Campbell, Dana Hughes, and Katia Sycara are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. {ajadhav20, danahugh}@andrew.cmu.edu, jcampbell@cmu.edu, sycara@cs.cmu.edu

update its ToM to recognize that person A and person B are aware of the victim in room 203.

The ASIST project works to provide artificial agents with ToM models to understand the behaviors of individuals and teams [4]. State-of-the-art language models may be trained on tens of terabytes of data [5]. Trials for the ASIST project are expensive, limiting the quantity of language data. Additionally, the speech of participants in the trials may be unclear, participants may speak over one another, and participants may unintentionally mute their microphones, reducing data quality. Once data are collected, they must be labeled by hand, a time consuming process. Data augmentation can mitigate these issues.

II. RELATED WORK

A. Theory of Mind

Humans as well as chimpanzees are believed to have a ToM. If a ToM is not unique to humans, one may be more inclined to extend it to machines. This has been defined as:

An individual has a theory of mind if he imputes mental states to himself and others. A system of inferences of this kind is properly viewed as a theory because such states are not directly observable, and the system can be used to make predictions about the behavior of other. [1]

With a ToM, humans communicate efficiently and adapt their speech to the listener or listeners. As a result of this, a ToM is important for communicating with a human or team of humans, since it enables adaptation to the listener. A human recognizing and correcting a misunderstanding regarding a communication is one application of a ToM [2]. Human ToM models may reflect high-level beliefs, but their utility is measured by the ability to make predictions and plan. A machine theory of mind may use a different underlying process, but it can also be measured by prediction and planning capability. A machine ToM can be able to recognize false beliefs, like a human ToM [6].

B. Ad hoc Teams

Ad hoc teams are teams in which members do not have prior experience working together [7]. Prior work has identified difficulties in these teams.

Communications, achieving situational awareness, engaging in standard teamwork behaviors, and demonstrating leadership were listed as major problems by the majority of the subjects. Leaders of ad hoc organizations face difficulties in task

allocation, anticipating team members' actions, and anticipating team problems. Ad hoc team members commonly experienced communication problems (1) in knowing when to communicate updates to their team members (2) in knowing whom to ask for information (3) in providing and accepting feedback. [8]

If one has experience working with a partner, they may be able to make reasonably accurate predictions about the other's state of mind and actions. However, with limited experience, this becomes difficult. An agent may be able to supplement a human's limitations in understanding a state of mind through interventions. Prior work has identified agent interventions for ad hoc teams and identified the importance of communication in ad hoc teams. When one is unable to make estimate the state of mind of another, communication may be able to fill the gap. If one person makes an action that the observer is unable to explain, the observer might ask about the intent of this action. Additionally, one might inform others of their intents in an ad hoc team, since the other team members might not understand the actor's actions. These communications do not have to come from a human, however.

The four basic categories of possible interventions include: 1) agents that help the humans with basic task skills like navigation or providing user interface assistance; 2) agents that monitor task progress such as timekeeping or coverage monitoring. 3) agents that check whether all team members are fulfilling agreements on plan and roles related to the team search pattern; 4) agents that help humans with their teamwork skills such as regulating communications with teammates or alerting players to possible assistance opportunities. [7]

Participants are provided minimal training in the USAR environment. Participants may have difficulty with parts of the testbed, such as the map. In this instance, Intervention 1 may be useful. Agent interventions could also be used to fill gaps in a ToM by providing explanations for actions or recognizing errors in models.

In [7], ad hoc teams were studied where teamwork was more useful and where teamwork was less useful. When teamwork was less useful, it has been found that off-topic communication was harmful to performance. When teamwork was more useful, off-topic communication was found to increase score. One hypothesis is that this was a result of the off-topic communication warding off harmful communication. If an ASI agent can encourage or discourage certain types of communication, such as off-topic communication, it may be able to improve team performance. A difficult challenge of assisting teams has been: "how can team-supporting agents acquire a model of what the human team members intend to do and thereby be enabled to monitor their task execution and coordination as a team" [7]. A ToM model may be a good fit for this task.

C. Data Augmentation

When linguistic data are limited, the value of data augmentation increases. If communications must be hand-labeled, there may be an additional time-consuming process. Three common methods for augmenting linguistic data are:

- 1) Rule based
- 2) Interpolation
- 3) Model

These have been found to be in order of increasing data need [9]. When linguistic data are extremely limited, there may only be sufficient data for rule based augmentation. Two methods are Easy Data Augmentation [10] and Dependency Tree Morphing [11]. For systems that tagging communications, the increased classification accuracy of EDA may be useful [10].

Dependency Tree Morphing, which has been inspired from image processing, performs an equivalent of rotating and rotating on natural language. Cropping is a straightforward method, in which components are removed from sentences. Rotating involves moving fragments around the root of a dependency tree. These methods are believed to be useful for sentiment analysis and text classification, and with the natural language component wanting to understand the state of mind of participants and potentially label their communications, may be useful [11].

III. METHODS

A. Testbed

The ASIST testbed is a simulated urban search-and-rescue (USAR) scenario conducted in Minecraft. Ad hoc teams were formed of three participants, with one or more experimenters providing instructions and training for the teams. Participants connected to the testbed remotely, but were instructed to treat the test as if it were being conducted in a lab. Steps were taken to ensure regularity between teams, such as instructing participants to only use a single monitor, have an external mouse, and to use a similar proportion of screen space for the game, map, and interaction pane containing the marker semantics. The testbed uses a timer to provide a time pressure. The testbed uses two perturbations of a map, which differ in rubble and victim placement. The difficulty of each version is intended to be equal. Participants had screens recorded and spoke through Zoom, providing an audio recording.

B. Data Collection

Data are collected using Zoom recordings of participants completing trials remotely. Otter.ai, an automated speech recognition (ASR) service, is then used to generate a draft of the transcript. The ASR transcripts may contain errors, so a human listens to the transcript and corrects errors in speaker and in communication. This is especially important when participants do not speak General American English, as transcription errors appear to be more common. The transcript includes speaker name, utterance, communication number, and the start and end of the utterance with hundredth of a second.

C. Data Cleaning

ASR transcripts serve as a starting point of the NLP data. They may contain many errors, as well as complete omissions of utterances, especially when the voice of a participant is unclear or the volume is low. Data quality can be increased by having a human coder listen to the transcript and correct errors. By having higher quality data, it is hoped that NLP efforts will have more compelling results.

Transcripts are initially produced as a single file, including both trials from the testbed as well as additional data. Removing the components other than the trials and making a file for each trial makes working with the linguistic data easier.

The ASIST project involves multiple data collection sources, including a message bus. Having separate files makes aligning the message bus data based on time to the transcripts easier, since this involves looking at less data. Transcript data are aligned with the message bus based on the times of communications. This enables coded linguistic data to be merged with other data, such as participant location and team score.

D. Transcript Coding

The ASIST project works to build a theory of mind, so transcripts are given codes to potentially reflect state of mind. High level codes do not reflect the content of a participant's speech, instead, they include information like whether the speech is a question or if it is informing a teammate. Topic level codes involve the specifics of a communication, such as if it involves rubble or a victim.

Identifying speech at the level of communication, rather than at the level of content, is done. Content level analysis often requires additional data, and with the limitations in data collection, this becomes difficult. Communication level labeling enables analysis of other communication traits, such as the use of closed-loop communication.

E. Data Augmentation

Data collection in the Minecraft environment is difficult. To compensate for limitations in ASR data quantity, synthetic utterances were generated. Transcripts, after being corrected, were run through SpaCY [12] and the Berkeley Neural Parser [13-14]. SpaCY performs part-of-speech tagging tokenization, and segmentation, which increases the accuracy of the parser output. The parser was selected on the basis of high accuracy. Parsing is often used before rule based data augmentation. The parser outputs Backus–Naur form grammar, which enables words used in the same part of speech to be substituted for data synthesis. SpaCY is able to generate dependency trees, enables dependency tree morphing [10]. The parsing is used to prepare for data augmentation.

IV. EXPERIMENT

A. Environment

In the scenario, three participants are tasked with rescuing as many victims as possible during two trials, each with different maps. At the entry point of the game, participants



Fig. 1: The semantics provided to two of the three participants.

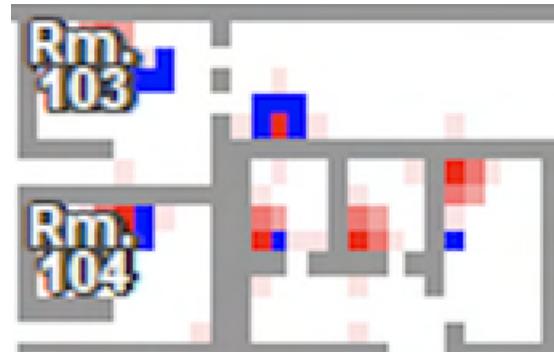


Fig. 2: Part of a heatmap displaying victim locations and room names.

are able to select one of three classes: searcher, medic, and engineer. The searcher has a stretcher to move victims, the medic has a medical kit to heal victims, and the engineer has a hammer to remove rubble. Each tool has durability and will wear out; once depleted, participants move faster. The searcher has the fastest movement speed, the medic's speed is in the middle, and the engineer moves the slowest. As the searcher is fastest, this class may be best able to find victims. Victims may be hidden by rubble, necessitating the use of an engineer (Fig. 3). There are two types of victims: critical and non-critical. Non-critical victims can be healed only by a medic. Critical victims must also be healed by a medic, with the additional requirement that all three participants must be near the victim for the revival process. Once a victim is revived, it remains in the state.

Participants are given marker blocks to help determine what rooms have been explored and what is inside them. They are labeled 1-3 and each participant has a different color. Two participants share a single provided semantics, with the third participant having the no victim and regular victim swapped (Fig. 1).

Participants are also given a heatmap including room names and victim locations. Participants share some components of the map, but a complete map is not given to each participant. This encourages participants to share knowledge of victim location. Participants are also given the opportunity to recognize that their maps contain different information, as no one participant has a complete heatmap. Part of a heatmap is shown in Fig. 2.

As each participant may be only one class, teamwork

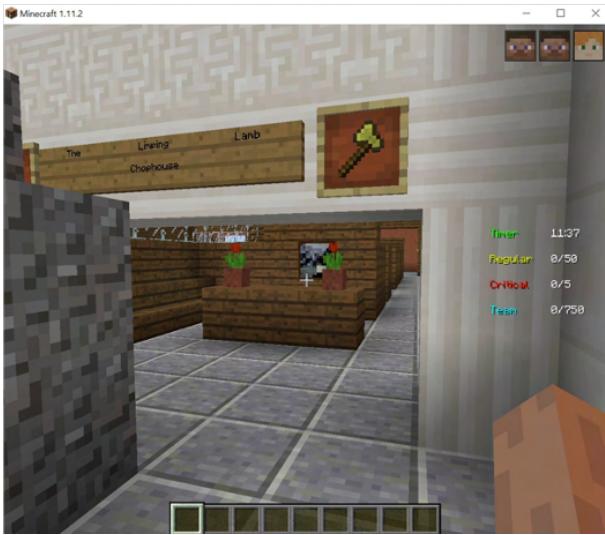


Fig. 3: A participant’s view of the Minecraft environment.

is encouraged. There is also a time limit, simulating the importance of speed in a disaster scenario, again increasing the importance of teamwork. Due to the utility of teamwork, participants are naturally encouraged to communicate. These communications are recorded and analyzed to identify salient features.

B. Data Collection

Data were collected using automatic speech recognition (ASR) on human trials in a search-and-rescue environment. These transcripts were hand-labeled with high-level and low-level, or domain-specific, language (Table 1). Imbalance in code frequency may be addressed by over sampling and under sampling [7].

Zoom ASR transcripts may excessively break down communications; for example, a single sentence might be broken into multiple utterances. This is not desired, so the high-level code Fragment is used so a program can recognize when multiple utterances are really fragments of a single utterance. The first utterance is always labeled with the non-fragment code, and all subsequent fragments are labeled with the fragment code.

High-Level Codes	Low-Level Codes
Question	No code
Command	Rubble
Inform	Victim
Agreement/Closed-loop	Marker
Intent	Role
Other	Tool
Fragment	

TABLE I: Communication Codes

C. Data Cleaning

Steps were taken toward a transcript coding standard to improve inter-rater reliability. This begins with standardizing software and enumerating each step of the process. Table 2

describes the steps of coding transcripts. It is suggested that the coder has a copy of the map to ensure locations are heard correctly.

- 1) Make an Excel Workbook named team(number).xlsx and select cell A1. Open the Zoom transcript, select all, and paste into Excel.
- 2) Make a copy of the current sheet and put it at the end of the Workbook.
- 3) Cut off ends of the transcript that are unrelated to the mission. The “321” countdown should be left in the transcript. The beginning of the transcript should have the communication number. Transcript should end with the last communication of the mission.
- 4) Add column names: Transcript, Corrections, Codes (for high and low level codes), Predictable (if codes can be predicted only from the current utterance, Coding_Suggestion (if, from this utterance, the coder suggests to add a high or low level code).
- 5) Copy all cells of the transcript column into the corrections column. Will need to rename the corrections column.
- 6) Correct transcription errors while listening to Zoom recording. May complete step 7 simultaneously.
- 7) Add communication codes.
- 8) Now that coding is complete, export the sheets to csv with the naming scheme: team(number)_trial(number).csv.

Step 3 aids in working with other data sources. Metadata files may begin timing with the start of a mission, however, Zoom recordings include additional communications. The ASIST agent only uses data from trials, so removing the excess communications will help with future training and use of Zoom data with trial metadata. The spreadsheets are exported to comma separated values files for better compatibility with scripts. Additional standardization was used to improve inter-rater reliability.

- Each trial is 15 minutes and they appear towards the middle to end of the transcript
- The beginning of each trial is announced with “321” Each transcript concludes with the end of the second (of two) trials
- If transcript does not record the speaker name, it should be added to the transcript if the speaker can be determined by the coder
- Question marks should be added to questions in the transcript; these are not included in Otter.ai transcripts
- If an utterance is not recorded, it should not be added to the transcript, as this often occurs when a participant’s speech is unclear
- Capital letters should be used when participants reference rooms for increased transcript readability
- “Okay” is preferred to “OK,” in consistency with Otter.ai transcripts
- If participant A speaks in utterance N and B in N+1, but some of B’s words are in N, B’s words should be moved to utterance N+1

- Overlapping communications, in which the coder cannot determine the topic, should not receive a code
- Communications between participants and those conducting the test should not receive a code

D. Data Alignment

The Zoom transcripts contain only speech data, with the message bus containing additional data including team score. If a task is to predict team score based on speech, data for this task can be collected by combining the annotated transcripts with the message bus data. The following steps convert 15:03 on the metadata timer to the Zoom transcript time.

- 1) Open the video of naming scheme HSR-Data_OBVideo_Trial-T0*.mp4
- 2) Look at the video for when the timer is at 15:00
- 3) Find the first utterance that appears in both the Zoom transcript and is heard in the video. Subtract the number of seconds this occurs after the timer reaches 15:00 in the video and then subtract three seconds
- 4) Take the time from the previous step to get the time that the trial started from the beginning of the Zoom transcript

V. RESULTS

After parsing, the grammar of human utterances is extracted. This includes the word, grammar type from Backus-Naur form syntax, and frequency. With frequency, data may be generated as a stochastic process, with more common words in human utterances appearing more frequently in synthetic utterances. Parsing is conducted in SpaCY, so the code may be easily extended to use SpaCY’s part of speech tagging for data augmentation. With the parsed and coded transcripts, data synthesis can begin with little additional work.

VI. DISCUSSION

Future natural language processing efforts may be improved by increasing the sound quality of recordings, enabling more accurate ASR transcripts. With more accurate transcripts, recognition and prediction of participant action may be less challenging. Instructing participants to not talk over one another, while reducing the authenticity of participant behavior, may improve transcript quality.

Finally, a decision could be made on whether transcripts should be coded while the coder listens to the audio recording. Transcripts may be missing utterances, so communication codes may reflect unrecorded utterances when coding at the same time as listening to the recording. If coders do not have a standard, coded transcripts may have increased variation between coders.

Eventually, an ASI agent may be used to provide feedback. If participants have a false belief, the agent would need to recognize this and inform the participants in a method they are receptive to. In the ASIST testbed, participants are given different marker semantics. An ASI agent might intervene by encouraging participants to use a standard, rather than the conflicting semantics they are provided.

ACKNOWLEDGMENTS

The authors would like to thank Rachel Burcin, John Dolan, the RI community, and the mentors and sponsors of the Robotics Institute Summer Scholars program. This work was funded by the Carnegie Mellon University Robotics Institute.

REFERENCES

- [1] D. Premack and G. Woodruff, “Does the chimpanzee have a theory of mind?” *Behavioral and Brain Sciences*, vol. 1, no. 4, p. 515–526, 1978.
- [2] H. Zhu, G. Neubig, and Y. Bisk, “Few-shot language coordination by modeling theory of mind,” *CoRR*, vol. abs/2107.05697, 2021. [Online]. Available: <https://arxiv.org/abs/2107.05697>
- [3] C. Frith and U. Frith, “Theory of mind,” *Current Biology*, vol. 15, no. 17, pp. R644–R645, Sep 2005. [Online]. Available: <https://doi.org/10.1016/j.cub.2005.08.041>
- [4] “Artificial Social Intelligence for Successful Teams.” [Online]. Available: <https://www.darpa.mil/program/artificial-social-intelligence-for-successful-teams>
- [5] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” *CoRR*, vol. abs/2005.14165, 2020. [Online]. Available: <https://arxiv.org/abs/2005.14165>
- [6] N. C. Rabinowitz, F. Perbet, H. F. Song, C. Zhang, S. M. A. Eslami, and M. Botvinick, “Machine theory of mind,” *CoRR*, vol. abs/1802.07740, 2018. [Online]. Available: <http://arxiv.org/abs/1802.07740>
- [7] G. Sukthankar, K. Sycara, J. Giampapa, and C. Burnett, “Communications for agent-based human team support,” pp. 285–313, 01 2009.
- [8] R. Pascual, “Supporting distributed and ad-hoc team interaction,” in *International Conference on People in Control (Human Interfaces in Control Rooms, Cockpits and Command Centres)*. IEE, 1999. [Online]. Available: <https://doi.org/10.1049/cp:19990164>
- [9] S. Y. Feng, V. Gangal, J. Wei, S. Chandar, S. Vosoughi, T. Mitamura, and E. Hovy, “A survey of data augmentation approaches for nlp,” 2021.
- [10] J. Wei and K. Zou, “Eda: Easy data augmentation techniques for boosting performance on text classification tasks,” 2019.
- [11] G. G. Şahin and M. Steedman, “Data augmentation via dependency tree morphing for low-resource languages,” 2019.
- [12] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd, “spaCy: Industrial-strength Natural Language Processing in Python,” 2020. [Online]. Available: <https://doi.org/10.5281/zenodo.1212303>
- [13] N. Kitaev, S. Cao, and D. Klein, “Multilingual constituency parsing with self-attention and pre-training,” in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, July 2019, pp. 3499–3505. [Online]. Available: <https://www.aclweb.org/anthology/P19-1340>
- [14] N. Kitaev and D. Klein, “Constituency parsing with a self-attentive encoder,” in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 2676–2686. [Online]. Available: <https://www.aclweb.org/anthology/P18-1249>

Trajectory Planning for a UAV Wrench Task Considering Vehicle Dynamics and Force Output Capabilities

Andrew Ashley¹ Azarakhsh Keipour² Sebastian Scherer²

Abstract—As unmanned aerial vehicles (UAVs) have risen in popularity, so has the interest in using these vehicles to physically interact and manipulate the world around them. From inspecting metal thickness and quality in hard-to-reach areas to delivering packages, the demand for the use of UAVs has outpaced traditional methods of trajectory planning for these vehicles. Traditionally, methods rely on kinematic and geometric models [1] in order to plan trajectories, but these methods fail to take into account the constraints and forces that physical interaction introduces. Each multirotor's geometry and architecture create an operational profile with constraints and patterns specific to the robot. The operational profile includes the limits (such as the maximum wrench and maximum tilt) and defines the possible range of outputs in each state. There has been little exploration into the addition of these operational profiles to traditional planning methods, and as such, the current state-of-the-art methods fall short of efficient and robust planning systems for a UAV interacting with the world. This work explores developing an operational profile of a fully-actuated, 6 Degree of Freedom (DoF) hexacopter and integrating it into a trajectory generator that uses the operational profile and a defined task to provide motion plans that are informed by the vehicle's dynamics. This work also introduces a motion controller strategy that is adapted to follow motion-wrench trajectory plans that will allow the vehicle to execute the manipulation task. The new methodologies introduced in this paper bridge the gap in traditional planning systems that are made for free-flight planning to allow UAVs to successfully interact with the world around them.

Index Terms—Aerial Systems: Perception and Autonomy, Field Robots, Task and Motion Planning

I. INTRODUCTION

UAVs have the ability to easily reach places that would be very difficult for humans or other ground robots to get to. This advantageous maneuverability provides countless opportunities for an aerial robot to complete tasks and manipulate the world in safer and more efficient ways than have been previously possible. One example of such a task is the rotation or screwing of a target object in a high-to-reach area. The development of omnidirectional UAVs and control systems for these vehicles has brought this task into reach, and this work proposes a method for planning a set of trajectories for a UAV to perform a wrenching motion on a target. In order to plan a set of trajectories for a UAV that is providing a wrench force with a fixed end effector on an object, the planning system must be informed of the vehicle's dynamics and capabilities.

¹Andrew Ashley is with the University of Pittsburgh, United States
ata33@pitt.edu

²Azarakhsh Keipour and Sebastian Scherer are with Carnegie Mellon University, United States akeipour@andrew.cmu.edu
basti@andrew.cmu.edu

II. RELATED WORK

A. Efficient Sample-Based Planning for High-Dimension Spaces

A high volume of work has been completed on efficient planning algorithms for trajectory generation. The Open Motion Planning Library (OMPL) has implemented many of these algorithms and allows for relatively quick development of planning systems [2]. Due to large multiple-dimension space that the uav system needed to plan in, a sample based planner was optimal due to its relative speed over complete planners and the lack of need for a provably optimal solution. RRT*-Connect, explained further in Section 3.D.1., was chosen from OMPL for this work.

B. Trajectory Planning Considering Vehicle Dynamics and Force Application

In the desired wrench task, the planning system will need to plan for movement in both X,Y,Z space as well as the roll and pitch angles of the UAV. This system also needs to have some form of temporal constraint that only allows the vehicle to start rotating/providing wrench force once it is in contact with the target rotation point. The work done in [3] on an underwater submarine with a robotic arm used different Motion Primitives to coordinate movement between the submarine and the arm based on distance to the target. This system only planned to move the arm once it was a certain distance from the target. This system provided the inspiration of the multi-space planning proposed in this paper. While completing a wrench task, the manipulator may experience a wide range of varying necessary forces that need to be overcome in order to rotate the target. [4] has completed work that uses a search based planning system that considers the manipulator's torque constraints and dynamics in order to plan feasible trajectories of the manipulator based on the maximum torque that each portion of the manipulator can output.

III. METHODOLOGY

A. UAV Platform for Experiments

The vehicle to be used in the development and testing of the proposed approach is a fully actuated 6 DoF hexacopter Fig. 1 simulated in Gazebo simulator. As seen in Fig. 2, the rotors of the vehicle are all mounted at +/-20 degrees off the standard vertical thrust orientation. This, paired with the controller introduced in Section 3, allows the vehicle to have 6 DoF's, with lateral movement unbounded by the roll and pitch of the vehicle, which is unique compared to standard multirotors.



Fig. 1. Fully-Actuated Hexacopter

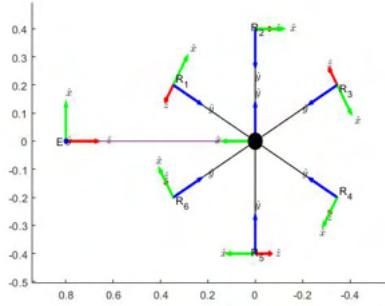


Fig. 2. Thrust Components of Fully-Actuated Hexacopter

B. Gathering Vehicle Configuration Dynamics and Creating an Operational Profile

Creating a vehicle dynamics informed planner first requires an analysis of the proposed vehicle in order to determine the maximum operational pitch and flight configurations for a required torque value. A simulator was first used to analyze the maximum angular acceleration at different roll angles around the roll axis at a fixed pitch angle (Triangle Figure). This was then generalized to create a formula that provides the maximum roll and pitch angle combination that can provide the required torque to rotate the target. This formula, along with the set of roll and pitch where

$$\text{ThrustOutput} > \text{MassOfVehicle} * \text{Gravity}$$

were combined to create a vehicle specific operational profile for the test UAV.

C. Fully Actuated UAV Controller

In a previous work, a popular flight controller, PX4, was extended with a controller for the fully-actuated hexacopter that is used for this work [5]. This controller provides multiple attitude and lateral thrust strategies to allow for the 6DOF drone to be used in applications such as the wrench task of this paper. One of these strategies, called the zero-tilt strategy, allows the vehicle to move translationally in $[x, y, z, \theta]$ while keeping the roll of the drone and the pitch of the drone zero, seen in Fig 4. This translational movement without roll or pitch allows the vehicle to make optimal flush contact with the target rotation point, which is critical to the

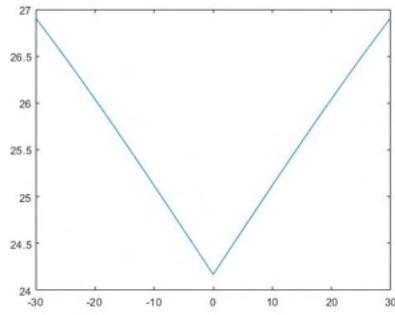


Fig. 3. Maximum Thrust Output For Varying Roll Angles with a Fixed Pitch

wrench task. Another ability of this controller and the fully-actuated drone is that it allows the vehicle to maintain a fixed $[x, y, z]$ position while changing roll angle. This facilitates the rotation of the drone while the end effector is in contact with the rotation target. For all of the experiments in this work, the zero-tilt strategy was used until the drone's end effector contacted the target, where the fixed attitude strategy was then used to rotate the drone and the target.

D. Planning System

1) *RRT*-Connect*: The planning algorithm used in this work is RRT*-Connect. This algorithm combines two earlier planning methods, RRT* and RRT-Connect and provides a single planner that combines the benefits of each. Standard RRT (Rapidly-exploring Random Tree) algorithms initialize a parent node at a start point and randomly sample nodes in the space and connect valid ones to the existing tree structure until it reaches the goal. RRT [6] was developed for planning in high-dimension spaces, which is ideal for the goals of this work. While this is useful, it does not guarantee an optimal path and is slow compared to new iterations of the method. One iteration, RRT* [7], uses tree rewiring and an added cost function to provide a theoretical optimal path as the number

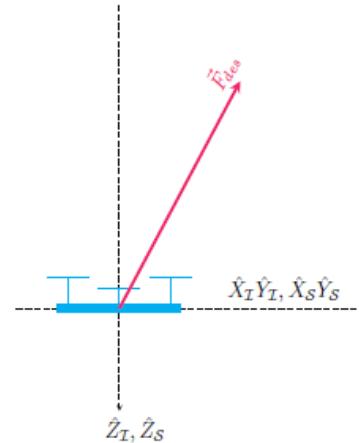


Fig. 4. Zero-Tilt Control Strategy

of nodes sampled goes to infinity. Another, RRT-Connect [8], improves on the speed of RRT by initializing trees at both the start and goal state and building both trees until they connect. RRT*-Connect builds upon two of these iterations, RRT* and RRT-Connect in order to provide a fast planner for high-dimension spaces that provides a theoretically optimal solution. RRT*-Connect was also readily implemented in OMPL, which further aided the speed of development.

2) *Hybrid State Space*: Planning to turn a targeted object with a 6DoF UAV introduces novel challenges that do not exist in classic trajectory planning for obstacle avoidance tasks. Since the vehicle's maximum wrench force around the end effector varies with different roll and pitch configurations, the planner must be aware of the system's dynamics and their relation to the known force that the task requires. The planner, given only a target position and orientation $[x,y,z,\theta,roll,pitch]$ must also be able to recognize that the vehicle should only rotate to the desired roll and pitch. This introduces a temporal aspect to the planning that must be solved.

This work overcomes this challenge by using online planning and a hybrid state space that includes both considerations of the vehicle's dynamics when determining the validity of each sampled state and temporal constraints to ensure rotation is only planned once the vehicle is in contact. The planning space in total covers the SE3 space: $[x,y,z,\theta,roll,pitch]$. Depending on the drone's configuration during planning, parts of the space are fully constrained to enforce correct planning with respect to time. While the vehicle is not in contact with the target, a R3 space $[x,y,z]$ was utilized to plan in the World Space while the SO3 space for roll pitch and θ was fully constrained to zero. Valid states in this configuration of the state space included all states that were not occupied by an obstacle in World Space and did not have a non-zero value for $[roll, pitch, \theta]$.

The sensor attached to the end effector of the vehicle informs the planner when contact with the target is made, which initiates the transition in the planner to plan in the full SE3 space, with the roll pitch and θ components unconstrained. In order to determine a sampled state to be valid in this space, first collisions with obstacles in the R3 World Space were checked. If those elements of the state were admissible, the roll, pitch, and θ components were input into the vehicle configuration dynamics formula, introduced in Section 3B, in order to determine if that configuration would be able to output a torque that was above the minimum required torque for the task. If both of these were viable, the trajectory was added to the RRT*-Connect's graph as a valid state.

IV. EXPERIMENTS AND RESULTS

The planning and control system proposed in this work was tested through a series of constrained experiments. The R3 portion of the hybrid state space used to represent physical obstacles was first tested without a rotation target in order to evaluate its effectiveness and accuracy. Once the simple obstacle avoidance task was fully tested, the SO3

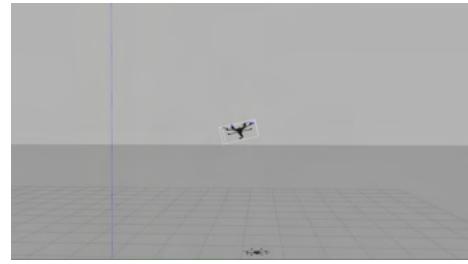


Fig. 5. Vehicle navigating to target



Fig. 6. Simulation test environment

portion of the hybrid space was tested. The vehicle was commanded to take off and given a task to rotate an imaginary target with a fixed torque requirement. The planner proved successful in this task, correctly considering the vehicle's maximum torque output during the task while rotating the imaginary target. After both portions of the planner were evaluated separately, they were combined in order to test the temporal constraints imposed on the planning system by the contact sensor. Because the goal of this test was solely to evaluate the planner's ability to output a trajectory that first navigates to the target and then once connected, rotates the target, no obstacles were used. For demonstration and visibility purposes, the physical representation of the target in the simulation was removed, although the mathematical constraints and representation of the target were still kept and used in the planning of the rotation of the target. This test can be seen in Fig 5 and Fig 6. In Fig 5, the planner output a set of trajectories to move the vehicle to the target rotation point. It successfully navigated to the point, and in Fig 6, the vehicle can be seen rotating the theoretical target at the goal position. Later tests were successfully conducted that tested both the obstacle avoidance and target navigation of the R3 portion of the hybrid planner, as well as the SO3 portion's ability to rotate the target while considering the vehicle's dynamics. The test environment can be seen in Fig 7.

V. CONCLUSION AND FUTURE WORK

As demonstrated in the previous section, the proposed hybrid planning system allowed the successful completion of a motion-wrench task on a target. The planner correctly navigated to the target point and then planned a correct series of rolls and pitches to successfully rotate the target. This

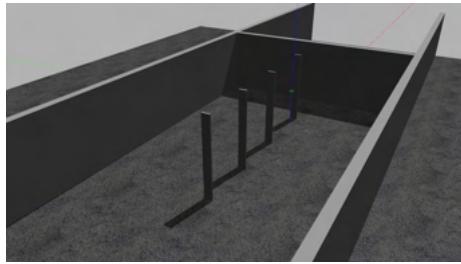


Fig. 7. Simulation test environment

work contributes the successful development and integration of a vehicle's force output constraints in the planning of a world manipulation task. The next step for this project involves integrating the developed software into an actual 6-DoF drone that has been built. The experiments will test a range of rotation tasks with varying force and torque requirements which will further evaluate the feasibility of this approach

ACKNOWLEDGMENT

The author would like to thank Carnegie Mellon University's RISS program for all of the opportunities for research and personal development that this summer research program has provided. They would also like to thank Rachel Burcin and Dr. John Dolan for their support and teaching. Finally, the author would like to thank Carnegie Mellon University's AirLab for the funding of this research and its members for support during this project.

REFERENCES

- [1] D. Brescianini and R. D'Andrea, "Computationally efficient trajectory generation for fully actuated multirotor vehicles," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 555–571, 2018.
- [2] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, December 2012, <https://ompl.kavrakilab.org>.
- [3] D. Youakim, P. Cieslak, A. Dornbush, A. Palomer, P. Ridao, and M. Likhachev, "Multirepresentation, mult heuristic a* search-based motion planning for a free-floating underwater vehicle-manipulator system in unknown environment," *Journal of Field Robotics*, vol. 37, no. 6, pp. 925–950, 2020. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21923>
- [4] J. Chuy, Oscar Y., J. Collins, Emmanuel G., A. Sharma, and R. Kopinsky, "Using Dynamics to Consider Torque Constraints in Manipulator Planning With Heavy Loads," *Journal of Dynamic Systems, Measurement, and Control*, vol. 139, no. 5, 051001, 2017. [Online]. Available: <https://doi.org/10.1115/1.4035168>
- [5] A. Keipour, M. Mousaei, A. T. Ashley, and S. A. Scherer, "Integration of fully-actuated multirotors into real-world applications," *CoRR*, vol. abs/2011.06666, 2020. [Online]. Available: <https://arxiv.org/abs/2011.06666>
- [6] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," Tech. Rep., 1998.
- [7] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the rrt*," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1478–1483.
- [8] J. Kuffner and S. LaValle, "Rrt-connect: An efficient approach to single-query path planning," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, 2000, pp. 995–1001 vol.2.

User Centered Approach for Developing a Robot Assisted Femoral Vascular Access Device for the Battlefield

Lama Bahanan¹ Nicolas Mateo Zevallos-Roberts² and Howie Choset²

Abstract—Femoral vascular access (FVA) is a critical life-saving intervention for managing trauma on the battlefield. There are many obstacles to performing FVA which automation could solve. Although there has been a substantial effort in developing automated vascular access technologies, none of them can be used for obtaining femoral vascular access. In addition, none of these technologies are designed for use in combat zones. In this paper, we use a user-centered approach to gather user requirements for developing a handheld robotically assisted femoral vascular access device tailored to military personnel regardless of their medical training. We conduct user interviews with medical staff from civilian and military settings to collect information about the clinical needs, implementation limitations, and design preferences. The findings from the user interviews were used to inform the design of the procedural workflow model of the device along with generating important criteria for the physical design of the device.

Index Terms—Medical Robots and Systems, Human Centered Automation.

I. INTRODUCTION

About 86% of battlefield deaths happen in the first 30 minutes of trauma [1]. This makes speed and accuracy the most important factors for trauma management [2]. However, due to the hostile environments a soldier can be left for more than 72 hours without medical care which increases the mortality rate [3]. In addition, providing medical care in combat zone encounters a lot of challenges such as limited availability of medical equipment and personnel [1]. For example, you may have the equipment for a procedure but no one to do it and vice versa. Another challenge is the combination of mental and physical stress along with the chaotic nature of a battlefield environment makes medical procedures even harder to perform [4]. Vascular access (VA) is crucial for trauma management. It is a prerequisite for fluid resuscitation, the most extensive and time-consuming block in the required basic combat training (BCT) for soldiers [4]. This precious time could be utilized for other lifesaving skills or tactical training [4]. All these challenges can be resolved by developing an automated vascular access device suited for use in battlefield conditions. Although a device capable of performing multiple VA methods would be optimal but developing the technology for an automated femoral vascular access would pave the way for other VA methods. There has been a huge advancement in incorporating medical

robots in the health care system. Various robotic systems have been developed for needle insertion to assist different vascular access methods [5], [6]. However, none are applicable for femoral vascular access [7]. In addition, there are no user centered studies for generating device requirements for vascular access devices for battlefield use. Conducting these studies is important as it has been found that obtaining user input in the early development stages increases the quality and usability of medical devices [8]. This paper's research supports the development of a robotically assisted femoral vascular access device [7] by collecting data from a user perspective that can be implemented in the design of the device. In this paper, we present an overview of vascular access, user centered design approach, user requirements for hardware features and the proposed workflow model that matches the expectations and needs of users in combat zones.

II. BACKGROUND

A. Femoral Vascular Access

Femoral vascular access (FVA) is one of the most common methods for obtaining central vascular access which is a method for entry and removal of devices or chemicals – blood, fluids, and medications - from the vascular system [9]. In addition, it is a prerequisite for advanced resuscitative methods such as Extracorporeal Membrane Oxygenation cannulation (ECMO) and Resuscitative Endovascular Balloon Occlusion of the Aorta (REBOA) [10].

B. User Centered Design

User Centered Design (UCD) approach is an iterative process in which the user is an active participant throughout the development phases of a product. UCD approach has been used widely for developing medical technologies ranging from devices to robots. In [8], they listed the benefits of utilizing a UCD approach which includes improving the efficiency, consumer approval, and usability. In addition, they summarized the methods for assessing user requirements during the different stages of medical device development such as conducting user interviews which is the main method in our study.

C. Related Work

There has been a substantial effort in developing robotic surgery technologies for vascular access. These technologies include the development of image-based guidance and needle insertion. In [6] they developed a device for cannulation of upper-extremity peripheral vessels and in [5] they developed a manipulator for central venous catheterization. However,

¹Lama Bahanan is a biomedical engineering student at Georgia Institute of Technology, Atlanta, GA 30332, USA lama.bahanan@gatech.edu

²Nicolas Mateo Zevallos-Roberts and Howie Choset are with the Biorobotics Lab, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA nzevallo@andrew.cmu.edu

none of these robots are suitable for performing femoral vascular access due to the unique anatomy of the femoral region [7].

Using a user-centered design is widely used in robotics. In [11] they utilized a UCD approach for developing a robot-assisted fracture surgery. They gathered user requirements through interviews with orthopedic surgeons which guided them through their robotic system development. In another study, they used the same approach for designing a prosthetic hand [12]. Nonetheless, there are no studies that investigate the user requirements for an automated femoral vascular access in combat zones. In summary, although there has been a lot of technological development with respect to the development of a robotically assisted FVA system [7]. There is a need for a parallel study into user design preferences for steering the hardware and software features to match the expectations of targeted users.

III. APPROACH

We conducted an extensive literature review in femoral vascular access, medical devices for the battlefield, and automated vascular access technologies which some of it can be seen reflected in the previous section. As a result, we devised the following objectives:

- Refine the concept for the new device.
- Identify barriers to safe design and implementation.
- Collect user opinions on hardware and software features.
- Develop a procedural workflow model.

To collect user opinion, we conducted user interviews with medical personnel with civilian and military experience from the United States. Each interview lasted between 20 minutes to 60 minutes. Before the interview, the goals of the study were clearly defined for the participants. During the interview, we ensured that the participants could talk freely, and additional questions were asked to clarify and encourage participants to elaborate more on their answers. Based on the research objectives we created the following clusters of questions:

- Needs: clinical needs and challenges while performing FVA.
- Barriers: Limitations that could affect the success of the procedure.
- Safety: Important factors to consider for ensuring the safety of the patient and operator.
- Design preferences: Understanding the users' preferences and environment.

During the interview, we showed the participants a video of the current device and graphical user interface (GUI) in action along with the proposed design of the device modeled using SOLIDWORKS and developed GUI using MATLAB to get their opinion about certain hardware and software features.

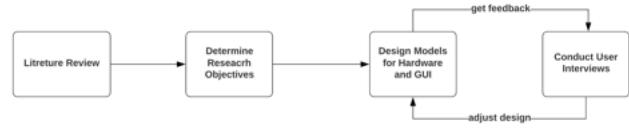


Fig. 1. The Iterative process in which our designs were adjusted as a results of feedback from user interviews.

IV. RESULTS

A. Hardware design considerations

There are four main important criteria for the hardware design: size, weight, portability, and stability. As the combat medic stated during the interview, the golden rule for developing any portable device used in the military is to have minimum size and weight and maximum power so it doesn't limit their deployability. Another important feature: stability rose when we asked the participants whether they prefer a single or two-handed design. Stability is an important factor while performing femoral vascular access. Abrupt motion during the needle insertion process could pierce the vessel and cause serious complications. Therefore, the handheld device must be designed ergonomically to ensure maximum hand stability and control.

B. Procedural Workflow

The device has to be designed to accommodate two types of users, femoral vascular access experts and nonexperts. For this reason, we created two modes - expert and nonexpert - with the same functionality, the only difference is that the nonexpert mode is supported with additional guidance throughout the procedure. In this paper, we present the procedural workflow for the nonexpert mode as shown in figure 2. In addition, the main screen has two default settings for bright and dark modes. Although more specific brightness options are included but this easy access is provided because light can be a source of attention in combat environments. In addition, there is a needle grip controller for changing the needle after each procedure.

Pre scanning: the first step is determining the site of procedure whether it is the left or right femoral region.

During scanning: a picture of where the operator should position the transducer will appear. The operator will be instructed through animation how to move the transducer along the desired region. The screen display will contain a reference image of how the ultrasound image should look like. Since the device is handheld, different operators will exert a variable amount of force. The amount of pressure impacts the quality of the ultrasound image. Therefore, there is a need for pressure guidance. This issue can be resolved by providing feedback to the operator through a feedback loop which will be demonstrated on the screen by a pressure scale. However, this is not an ideal solution because it can increase the cognitive load on operators. Therefore, integrating a force-controlled ultrasound system as developed in [13] is important.

After Scanning: Once the algorithm finds the vessel a red circle will appear around the targeted vessel and a needle overlay will appear on the screen. the ultrasound image will auto label vascular structures. Then the operator will be prompted to move the device until the tip of the needle overlay is on the center of the highlighted vessel. Once the system detects an overlap it will notify the user to hold the device still. There will be a threshold for movement which halts the needle insertion process until the motion is decreased to a safe level. Another safety feature is ensuring that we selected the right vessel before needle insertion by using the doppler ultrasound feature to check the flow which enables us to identify a vein from an artery. After needle insertion, a flexible guidewire will be inserted automatically. Once it is done, the operator will confirm the completion of the guidewire insertion which will retract the needle after which the operator can move the device without risking piercing the vessel which flags the end of the femoral vascular access procedure.

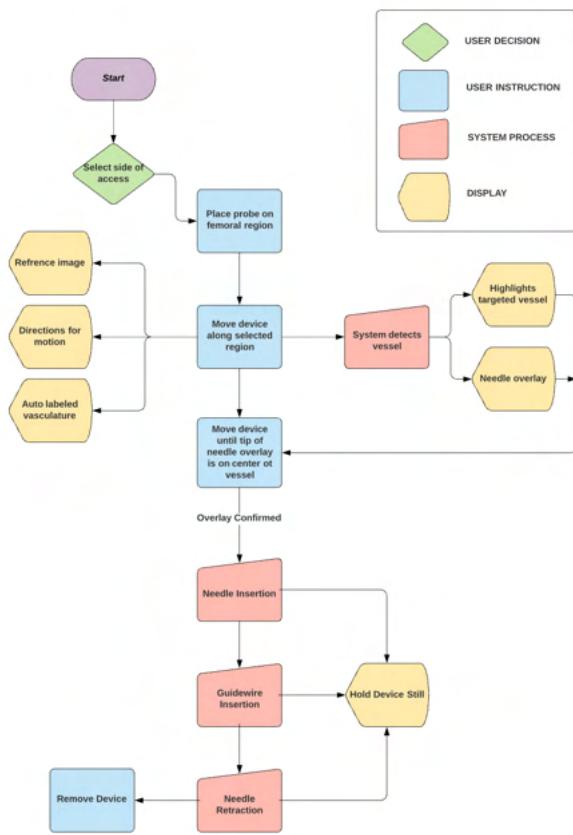


Fig. 2. Proposed workflow based on user requirements for the robotically assisted femoral vascular access device.

V. CONCLUSIONS

We were able to identify crucial hardware and software user requirements, but we had a limited number of interviews with medical personnel from different departments in the civilian and military settings due to their increased workload

because of the COVID-19 pandemic. In the future, we plan to conduct additional interviews to generate a more specific set of functional and nonfunctional requirements.

ACKNOWLEDGMENT

This work was supported by the Robotics Institute Summer Scholars Program and the KAUST Gifted Student's Program. Special thanks to Rachel Burcin and Dr. John M. Dolan for their continuous support. Also, I would like to thank Nicolas Zevallos, Even Harber and Dr. Howie Choset from the Biorobotics Lab at Carnegie Mellon University for their mentorship.

REFERENCES

- [1] J. Chu, "A robomedic for the battlefield," Apr 2020. [Online]. Available: <https://www.technologyreview.com/2009/02/03/95456/a-robomedic-for-the-battlefield/>
- [2] E. J. Hulse and G. O. Thomas, "Vascular access on the 21st century military battlefield," *J R Army Med Corps*, vol. 156, no. 4 Suppl 1, pp. 385–90, 2010.
- [3] M. Cox, "Army medicine embracing robot surgery, other high-tech procedures," May 2018. [Online]. Available: <http://www.military.com/defensetech/2018/05/14/army-medicine-embracing-robot-surgery-other-high-tech-procedures.html>
- [4] R. L. Mabry and P. J. Cuenca, "Should we teach every soldier how to start an iv?" *Mil Med*, vol. 174, no. 6, pp. iii–v, 2009.
- [5] Y. Kobayashi, J. Hong, R. Hamano, K. Okada, M. G. Fujie, and M. Hashizume, "Development of a needle insertion manipulator for central venous catheterization," *Int J Med Robot*, vol. 8, no. 1, pp. 34–44, 2012.
- [6] A. Chen, M. Balter, T. Maguire, and M. Yarmush, "Deep learning robotic guidance for autonomous vascular access," *Nature Machine Intelligence*, vol. 2, pp. 104–115, 2020.
- [7] N. Zevallos, E. Harber, Abhimanyu, K. Patel, Y. Gu, K. Sladick, F. Guyette, L. Weiss, M. Pinsky, H. Gomez, J. Galeotti, and H. Choset, "Toward robotically automated femoral vascular access," *ArXiv*, vol. abs/2107.02839, 2021.
- [8] J. L. Martin, E. Murphy, J. A. Crowe, and B. J. Norris, "Capturing user requirements in medical device development: the role of ergonomics," *Physiol Meas*, vol. 27, no. 8, pp. R49–62, 2006.
- [9] S. Bangalore and D. L. Bhatt, "Femoral arterial access and closure," *Circulation*, vol. 124, no. 5, pp. e147–56, 2011.
- [10] M. Chonde, J. Escajeda, J. Elmer, C. W. Callaway, F. X. Guyette, A. Boujoukos, P. L. Sappington, A. J. Smith, M. Schmidhofer, C. Sciortino, and R. L. Kormos, "Challenges in the development and implementation of a healthcare system based extracorporeal cardiopulmonary resuscitation (ECPR) program for the treatment of out of hospital cardiac arrest," *Resuscitation*, vol. 148, pp. 259–265, 03 2020.
- [11] I. Georgilas, G. Dagnino, P. Tarassoli, R. Atkins, and S. Dogramadzi, "Robot-Assisted Fracture Surgery: Surgical Requirements and System Design," *Ann Biomed Eng*, vol. 46, no. 10, pp. 1637–1649, Oct 2018.
- [12] V. Dimitrov, N. Cebry, c. Önal, and T. Padir, "Towards user-centered design of a robotic prosthetic hand with emg control interfaces," in *Proceedings of the 8th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, ser. PETRA '15. New York, NY, USA: Association for Computing Machinery, 2015. [Online]. Available: <https://doi.org/10.1145/2769493.2769536>
- [13] M. W. Gilbertson and B. W. Anthony, "Ergonomic control strategies for a handheld force-controlled ultrasound probe," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 1284–1291.

Multi-agent Coordination for Automation of U.S. Air Force Installed Systems Testing

Lauren J Blanks¹, Isaac K Isukapati²

Abstract— Complete end-to-end testing of installed systems on an aircraft within a free-space radio frequency (RF) environment is crucial to understanding and developing its combat capabilities. The Benefield Anechoic Facility at Edwards Air Force Base is one of the few facilities capable of measuring far-field coverage patterns of systems under test, including simultaneous measurements of multiple installed antennas for most aircraft in the Air Force inventory. However, the current test environment requires human intervention to manually re-position and reorient the emitters between discrete test points. The task of reorientation is very cumbersome and time-consuming, resulting in inefficient test operations. Additionally, this method only allows for snapshot characterizations of the aircraft's response since the manual set-up only allows for the test to be run in the specific and individual scenarios. Advances in distributed computing and multi-agent coordination techniques make it possible to not only automate the task of reorienting the emitters between test points, but also to simulate a continuous testing environment. As a first step in this process, this paper seeks to demonstrate the possibility of designing an autonomous process through the design of an auction-based scheduler capable of generating optimal tour for a given set of rovers and user-defined waypoints that they ought to traverse. We propose to benchmark the performance of the optimized system with the original test sequences to demonstrate the increased efficiency.

I. INTRODUCTION

The development and integration of installed systems on aircraft relies heavily on operational realism during testing to accurately emulate their combat capability and performance. This need results in increased emphasis on the use of specialized test facilities and flight test ranges [1]. Flight tests, however, present sufficient deficiencies to include significant cost factors due to the specialized environment required and the inability to enact controlled and repeatable tests [1]. Therefore, the testing of aircraft subsystems has increasingly relied on the use of ground test facilities capable of simulating a flight environment called Installed System Test Facilities (ISTF) [2]. One such ISTF is the Benefield Anechoic Facility (BAF) at Edwards Air Force Base, among the largest anechoic chambers in the world, capable of testing virtually all of the U.S. Air Force's inventory in a free-space radio frequency (RF) environment.

Improvement in test and evaluation (T&E) strategies at the BAF and similar facilities is paramount to ensure their ability to meet increased demand. Advancement efforts have been made to increase efficiency, such as the transition to

automated data acquisition and analysis methods during the test process at the BAF [3]. To assist in the efficiency of the testing process, the BAF now utilizes carts that can carry the emitters quickly from one test set-up to the next [2]. However, even this improved testing procedure requires manual intervention to re-position and reorient the antennas used to emit the electronic warfare (EW) signals during testing. As a result, the testing process is inefficient and requires substantial effort to conduct simulated tests, an increasingly apparent obstacle to the facility's mission as the demand for its use grows. Additionally, the static positioning of the emitter antennas inhibits the ability to conduct dynamic testing, as the requirement for manual positioning means that tests can only create snapshot characterizations as opposed to enacting a flying scenario, e.g. system response to signal on the flying approach.

Recent advancements in distributed computing and multi-agent coordination techniques indicate the possibility of automating the testing process to allow for reorientation and positioning between test points, to include simulating a continuous testing environment, more closely emulating a dynamic flight environment. In this way, this paper seeks to serve as a first step through the development of an auction-based scheduler that optimizes the tour for a given team of rovers equipped with the testing emitters. In comparing the performance of the simulated testing with the team of rovers with that of the original sequences, it is apparent that the testing is much more efficient and therefore, would allow the facility to better fulfill its mission.

II. METHODOLOGY

The methodology evolved from generating a graph and sequence of tasks for the rover team to complete.

In order to reasonably generate the graph and scheduler, assumptions were made that must be understood in order to establish the methodology of the research. Primarily, we assume that the rovers on which the emitters are positioned are able to localize their position within the graph and can navigate the area. In this way, this paper does not detail localization or communication techniques on the hardware of the rovers themselves. Second, the simulator assumes a controlled environment without obstacles or interference, allowing for simpler path planning and optimization of the sequence of tests. Finally, it is assumed that each test will utilize every available rover for the duration of testing.

The methodology can be divided into the following steps:
A) Chamber Description, B) Graph Generation, and C)

¹Lauren Blanks is with the Electrical and Computer Engineering Department, U.S. Air Force Academy.

²Isaac Isukapati is with the Robotics Institute, Carnegie Mellon University.

Shortest Path Algorithm. The information in this section elaborates on each of the steps.

A. Chamber Description

The Benefield Anechoic Facility served as the model for the generation of the graph to be used in the creation of the path planner and scheduler. The chamber itself is 264-ft. long x 250-ft. wide x 70-ft. high in order to house any aircraft in the Air Force inventory [4]. To create the free space environment needed to conduct the electronic warfare testing, the ceilings, floors, and walls are covered in radar absorbing material (RAM) that work to minimize the reflections of signal [3]. The RAM on the floor is able to be reconfigured based on the emitter positioning in order to allow for suitable coverage, as suitable radar absorbing treatment is essential to ensuring the testing data is not skewed because of reflected energy [5]. The RAM is shaped and positioned in such a way that it absorbs the electromagnetic signals and transforms them into thermal energy to reduce interference with the simulation [5]. Thus, one of the primary constraints in designing the paths on which the rovers could traverse was ensuring it did not negate the validity of the testing by removing too much of the RAM from the floor of the BAF. In order to maintain acceptable absorption within the chamber, the pathways could not account for more than 10% of the overall area of the floor dimensions. In this way, we could ensure that the operations of the rovers did not interfere with the accuracy of the testing. Additionally, the rovers were understood to operate outside of the Quiet Zone near the aircraft, an area in which nothing but absorbent material could be placed to protect the aircraft itself. In choosing which aircraft to use with the simulation, we settled on a C-130 Hercules, which was large enough to account for the more difficult testing scenario with limited space, but not so large that the rovers were excessively confined in their movement. The dimensions of a C-130 Hercules are 97-ft. long with a 130-ft. wingspan. With our graph of a C-130 Hercules observing a Quiet Zone of 24-ft, the graph structure begins with the inner grid positioned 24-ft in all directions from the aircraft, and this distance constraint was then extended to the space between the grids and subsequently between the outside grid and the wall, with a minimum of 24-ft distance spaced between each. For a smaller aircraft, additional grids could be included for the rovers to cover, with the opposite effect being possible for a larger plane, assuming the paths continue to allow for at least 90% coverage by RAM. A model of the chamber housing a C-130 Hercules, overlaid with the grid structure, can be found in Figure 1.

B. Graph Design

The graph was generated by creating a series of nodes on two grids that the rovers would traverse on their path around the aircraft. Each node serves as a way-point in which every node's location within the chamber is known, and each path would be determined by mapping the sequence of nodes that the rover would encounter from its starting point to

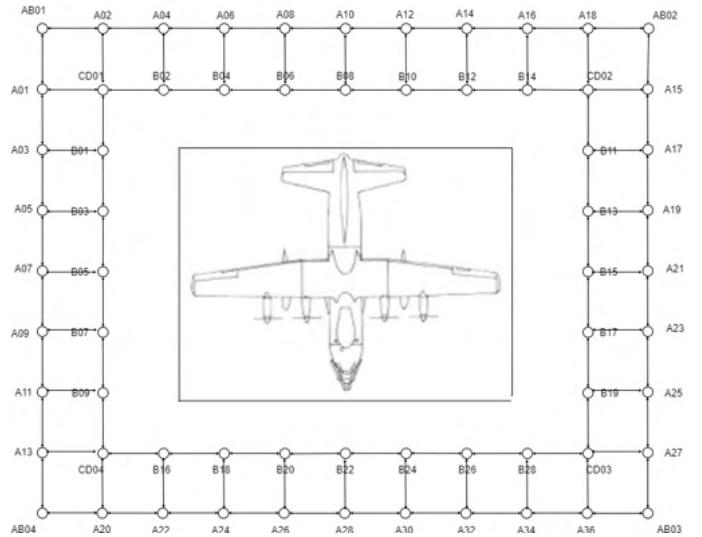


Fig. 1. Schematic of graph structure for C-130 in the anechoic chamber. C-130 image source: <https://fas.org/man/dod-101/sys/ac/c-130.htm>

final destination position for testing. Connecting each node within the same grid is an edge of length 24-ft., to allow for the rover to be able to accelerate to maximum speed and decelerate to full stop on a single edge. This length ensures the distance is enough to allow for the rover to slow completely, should a collision at the oncoming node need to be avoided. The outer grid consists of 32 nodes while the inner grid has 28, with uniform edge lengths of 24-ft. between nodes on the same grids and between nodes on neighboring grids. The grids and the connections between them offer the potential paths on which the rovers could travel prior to and during testing, so each edge is bidirectional to allow for traversal in multiple directions around the aircraft.

In order to understand the optimal path based on the kinematics of the rovers, it was essential to create a naming convention that would make it easy to identify when major transitions were taking place, such as encountering a corner or moving from one grid to the next. Therefore, as shown in Figure 1, the name of each node consisted of a sequence of either one or two alphabetical letters, followed by a number combination that indicated its position in the graph. The corners of each of the grids had two letters to indicate first that it was a corner, and second whether it was the inner or the outer grid. For example, the outer top left corner is called 'AB01', signaling it is the first corner on the outer grid, while the node at the same position on the inner grid is called 'CD01'. In the same way, nodes on the outer grid began with 'A' while nodes on the inner grid began with 'B'. The combination of numbers that follow the letters in the naming convention further inform the nodal position. Nodes residing in line with the x-axis as seen from the aerial view of the chamber shown in Figure 1, corresponded to evenly numbered nodes after the grid letter, while odd numbers corresponded to nodes parallel to the y-axis. In this way, it is clear where in the chamber a rover is currently, and

which direction it is heading based on the sequence of nodes that make up its path. For example, a rover with the path sequence: 'AB01', 'A02', 'A04', 'B02' travels from the top left corner horizontally on the outer grid for two edge lengths (16-ft.) before turning in to the destination of the node 'B02' on the inner grid. In this way, understanding the position and path of a rover would be much more intuitive and easier to follow.

C. Shortest Path Algorithm

Dijkstra's Shortest Path Algorithm was used to determine the optimal path for each rover to traverse from its source node, or starting point, to the destination node where it would be oriented for the tasks. Dijkstra's shortest path algorithm works to determine the optimal path by minimizing the cost from start to end on a positively-weighted graph [6]. The algorithm finds the shortest path by first finding the minimum cost connection between the starting node and its neighbors, thereby storing the connecting node in a set, S, of which the shortest, or minimum cost, length [7]. The same principle is applied at each connecting node until the destination is reached.

III. SIMULATION AND EXPERIMENTS

A. Graph Generation

The python package, NetworkX, was used in graph generation. First, we created a JSON file dictionary with the keys corresponding to each node on the graph, and their value pair was made up of their neighboring nodes. By importing that file into the graph code, NetworkX created a graph with the nodes and subsequently the edges connecting each to its listed neighbors. Following the creation of the graph as a NetworkX object, it allowed for the use of standard algorithms like shortest path to be implemented and used in analyzing the graph [8]. In this way, it allows for seamless use in the scheduling algorithm.

In order to generate a weighted the graph, it was important to attribute the cost, or time of traversal, to each edge. The rovers that will be used to carry the emitters are the Rover Robotics 4WD Rover Pro. The rovers themselves are capable of driving forward, backward, and pivoting at a stop. The maximum speed of the unloaded rover is 2.5 m/s and the acceleration is 1.5 m/s². The simplifying assumption was made that, due to the uniform length of the edges between the nodes and this knowledge of maximum speed and acceleration, the rovers would completely cross the connection between two nodes in 5 seconds. Therefore, every edge on the graph was given the weight of 5 seconds. From there, this knowledge of time to cross was incorporated in generating the graph.

B. Creating the Scheduler and Simulation

The layout for how the simulator was structured can be found in Figure 2.

First, random number generators were used to determine destination points for a series of tests. Random nodes were sampled from the graph and assigned to a rover as a

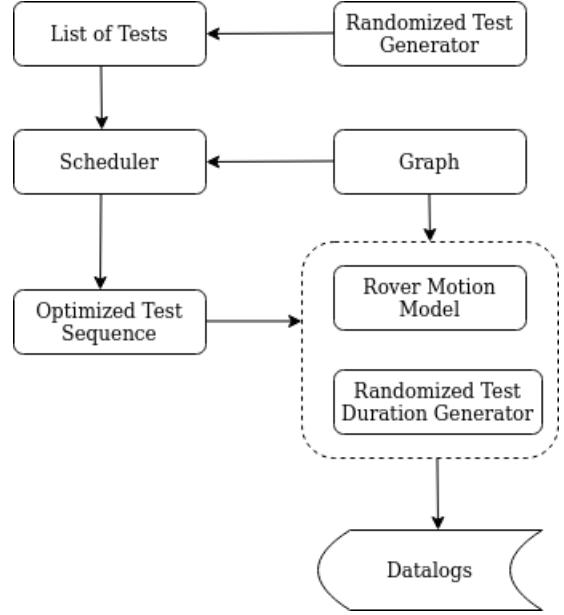


Fig. 2. Overview of the Simulator

destination for testing. Given four rovers in the chamber, four destinations were assigned per generated test, one for each of the rovers available. For this paper, we operated under the assumption that every test would utilize the maximum number of emitters and, thus, all four would be in use for each test. Upon the creation of each test's destination points, they were crosschecked with the already existing test points to ensure that no duplicate tests were conducted. Once cleared, the points were passed to the List of Tests, as shown in Figure 2.

Once a list of destination points for each rover was determined, it was sent to the auction-based scheduler, where the sequence would be optimized to minimize cost, or time of completion. In order to do this, each rover and its destination were analyzed through an algorithm catered to solving the traveling salesman problem. Inherently, each test had a different destination point for rover 1, and subsequently, rovers 2, 3, and 4. In this way, each was subject to its own case of the traveling salesman dilemma, where the scheduler would solve the optimal tour, or sequencing of the destination points, to minimize the time it takes to reach them all. First, the scheduler determined the fastest order of tests with rover 1's destinations as the basis, with the destinations of rovers 2, 3, and 4 simply lining up with whatever their destinations were for the tests that correspond to the optimal sequence for rover 1. For example, for tests numbered 1, 2, and 3, the optimal ordering for rover 1 to reach its destination nodes of the three tests the fastest might be to complete the tests in the order of 2, 3, 1. For this, because all four rovers are needed to conduct a test, rovers 2, 3, and 4, are confined to also navigate to the corresponding destinations for the tests in the order 2, 3, 1. For this reason, the scheduler produced four sequences, with each rover serving as the basis, meaning the sequence of the tests would be first determined by the

optimal order for rover 1, then it was created for rover 2, and so on through all four rovers. Then, the resulting four weighted sequences were compared, with the minimal cost, or fastest time, determined as the optimal sequence of the tests as a whole. The minimal cost sequence was the one that took the least amount of time overall to complete. In other words, the weighted cost of each sequence was actually how long it took the slowest rover in that sequence to reach its destinations and complete the tests. Sequence 1 might be optimal for rover 1, but that time was not what determined its cost. The cost was actually determined by the slowest tour for that sequence, as it is the limiting factor in pursuit of efficiency.

The optimized test sequence, informing the rovers not only which destinations they must navigate to, but also in what order, was then passed in with the graph information to be used in the rover navigation. The kinematic, or rover motion, model could then instruct the rovers on how to move and illustrate how quickly each sequence can be accomplished according to the acceleration profile. In this way, a time matrix was created that could describe the exact time each rover would arrive at the destination. An important addition at this stage was the random test duration generator, coupled with the rover motion model, as seen in Figure 2. At each destination, the rovers must wait the duration of the test in order to simulate the time it would take to sequence each test. The simulated test durations were generated using a Gaussian distribution with a mean of 90 sec. In this way, a final matrix could be passed to the data logs that not only listed the destinations for the rovers in the optimal sequence, but also the time that each rover would arrive at those destinations, and thus, the total time for completion of the generated list of installed systems tests.

C. Monte Carlo Simulations

In order to test the performance of the scheduler, 30 Monte Carlo simulations were run for each of nine different testing scenarios. The experimentation was grouped first by the number of rovers available for test, with the simulations run for a testing sequence involving either 2, 3, or 4 rovers. For each of these scenarios, three experiments were then conducted: the performance of the scheduler when sequencing 30, 50, and 100 tests. From there, the average time taken to complete the sequence of tests was recorded for each of the 30 simulations.

IV. RESULTS

The purpose of experimentation was to benchmark the performance of the scheduled sequence with the original. For each of the scenarios, sequencing 30, 50, and 100 tests for teams of 2, 3, and 4 rovers, the average time to complete the testing implementation was compared with the average performance of the unscheduled sequence after simulation. The side-by-side comparison of these averages can be seen in Tables I, II, and III. The optimized sequence performed the series of required tests anywhere from 5% to 8% faster than the original order. Additionally, the results of the experiment indicate that the more agents in the system, the better

the scheduler can improve the process. Figure 3 illustrates the cumulative distribution functions (CDFs) for the overall performance time for each of the nine experimental results, measured in seconds. These figures corroborate the results shown in the tables, as the optimized sequence consistently outperforms the original, by completing the task in a shorter duration of time.

The coupling of the integration of rovers with a scheduling algorithm capable of optimizing their tour would greatly improve the ability of the BAF to meet the rising demand for its test facility. Intuitively, the act of automating the test with the rovers removes the excess time typically taken to set up and take down the emitters' positions for each test sequence, and therefore, rover implementation is even more efficient.

TABLE I
EXPERIMENTAL RESULTS FOR 2 ROVERS

Number of Tests	Average Time of Completion (sec) :		Improvement
	Original	Optimized	
30	1746	1665	5%
50	3016	2843	6%
100	6127	5767	6%

TABLE II
EXPERIMENTAL RESULTS FOR 3 ROVERS

Number of Tests	Average Time of Completion (sec) :		Improvement
	Original	Optimized	
30	1768	1667	6%
50	3012	2812	7%
100	6144	5723	7%

TABLE III
EXPERIMENTAL RESULTS FOR 4 ROVERS

Number of Tests	Average Time of Completion (sec) :		Improvement
	Original	Optimized	
30	1767	1645	7%
50	3026	2785	8%
100	6151	5702	7%

V. CONCLUSION

In order to meet the current demand for realistic free-space simulation of aircraft, installed system test facilities like the Benefield Anechoic Facility must make their testing processes more efficient. One such improvement could be the automation of its testing through advancements in distributed computing and multi-agent system coordination to allow for a team of rovers to position and orient the emitters between and during installed systems testing. This paper serves as a first step towards that goal as an auction-based scheduler was successfully implemented to optimize the tour for a team rovers when given a set of waypoints and tasks. Future work could involve incorporating a realistic rover model rather than operating under the uniform weight assumption for the edge traversal time. Additionally, the scheduler could be bolstered to allow for more heterogeneous testing conditions, where a different number of rovers might be required for

each test to allow for optimization by performing some tests in tandem. Additionally, the scheduling algorithm could be further improved to solve the problem in greater dimension, where certain rovers were not specifically assigned nodal destinations that resulted in a TSP type of problem, but rather we could work to optimize the assignment of the rovers to test points based on current location in addition to the sequencing of the tests. The successful implementation of the auction-based scheduler allows for diverse and exciting ways to further improve and advance the automation of the U.S. Air Force's test and evaluation processes.

ACKNOWLEDGMENT

This research was supported by the United States Air Force Academy. Lauren Blanks would like to thank the USAFA electrical and computer engineering department, as well as Ms. Rachel Burcin and Dr. John Nolan for the opportunity to participate in RISS.

REFERENCES

- [1] E. F. Ali, "Electronic warfare testing at the benefitield anechoic facility," in *1997 IEEE Autotestcon Proceedings AUTOTESTCON'97. IEEE Systems Readiness Technology Conference. Systems Readiness Supporting Global Needs and Awareness in the 21st Century*. IEEE, 1997, pp. 232–243.
- [2] M. Pywell and M. Midgley-Davies, "Aircraft-sized anechoic chambers for electronic warfare, radar and other electromagnetic engineering evaluation," *The Aeronautical Journal*, vol. 121, no. 1244, pp. 1393–1443, 2017.
- [3] R. M. Taylor and J. Pasimio, "Automated data acquisition and analysis at the benefitield anechoic facility," in *1997 IEEE Aerospace Conference*, vol. 4. IEEE, 1997, pp. 139–148.
- [4] E. E. Sabat, "An integrated and collaborative radio frequency (rf) test infrastructure," AIR FORCE TEST CENTER EDWARDS AFB CA, Tech. Rep., 2012.
- [5] V. Rodriguez, "Basic rules for indoor anechoic chamber design [measurements corner]," *IEEE Antennas and Propagation Magazine*, vol. 58, no. 6, pp. 82–93, 2016.
- [6] J.-C. Chen, "Dijkstra's shortest path algorithm," *Journal of formalized mathematics*, vol. 15, no. 9, pp. 237–247, 2003.
- [7] N. Jasika, N. Alispahic, A. Elma, K. Ilvana, L. Elma, and N. Nosovic, "Dijkstra's shortest path algorithm serial and parallel execution performance analysis," in *2012 proceedings of the 35th international convention MIPRO*. IEEE, 2012, pp. 1811–1815.
- [8] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using networkx," Los Alamos National Lab.(LANL), Los Alamos, NM (United States), Tech. Rep., 2008.

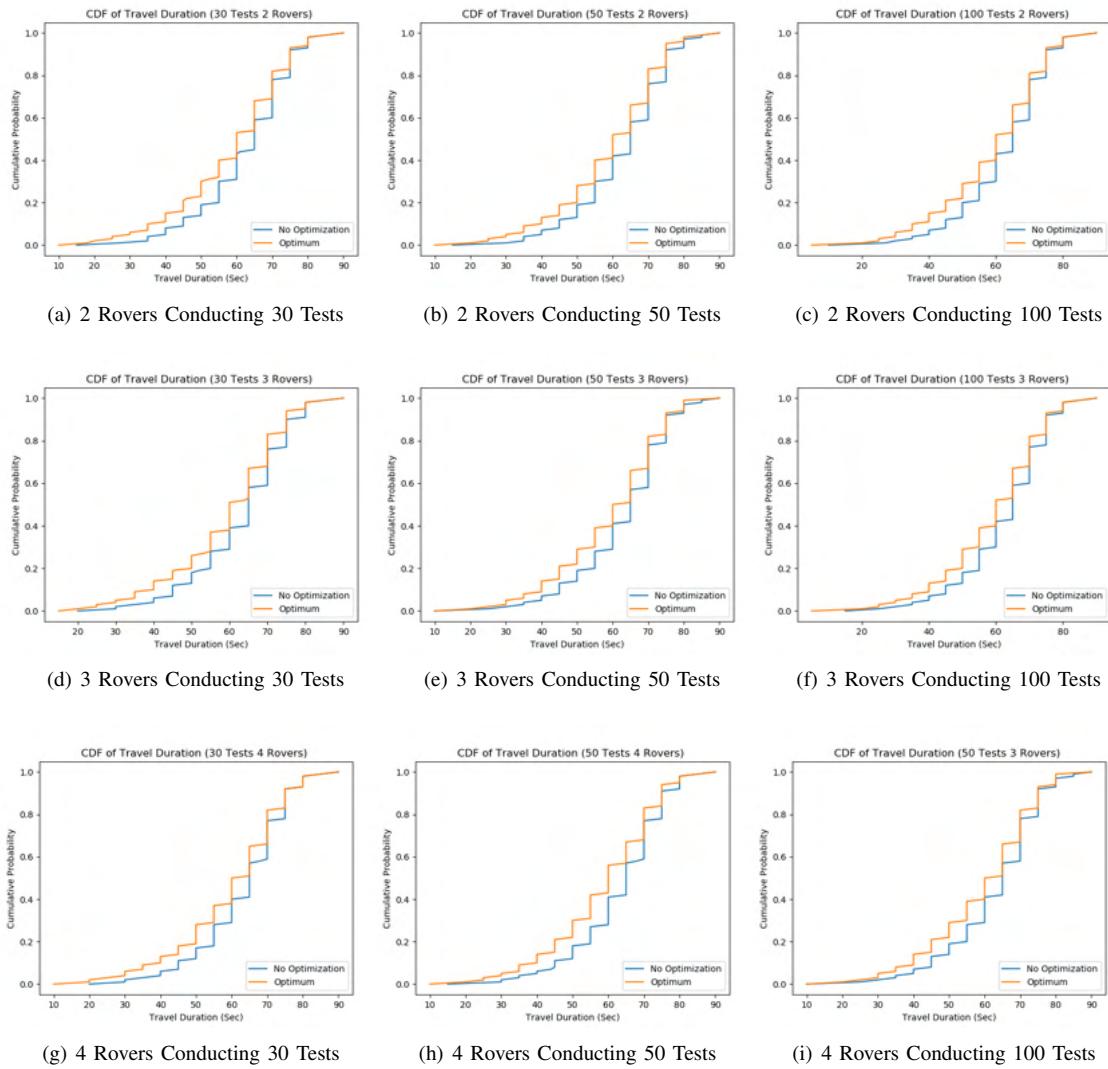


Fig. 3. Performance CDFs for systems with differing number of rovers and tests.

Growing Compliant Mechanisms From Mycelial Materials

Danielle Brennan¹, Keene Chin² and Dr. Carmel Majidi³

Abstract— Compliant mechanisms are flexible structures that achieve force and motion transmission through elastic body deformation. A mycelium is a dense network of thread-like structures of fungi. Mycelial (mycelium-based) materials provide an alternative to fossil-based plastics as they are completely biodegradable and renewable. Although many properties of mycelium are being researched heavily, research on the mechanical performance of such mycelial materials is limited. While mycelium is being introduced in the packaging and designing industries, little research has been done into mycelial materials' applications in active mechanisms, such as those used in medical devices or robotics. In this paper, we explore the mechanical properties of mycelium via mechanical prototyping and finite element analysis, in order to see if it is a suitable material for the construction of robotic mechanisms. Our analysis also aims to provide insight into other possible applications of mycelial materials.

Index Terms— Complaint Mechanisms, Mycelium

I. INTRODUCTION

Non-renewable materials are slowly being replaced by natural bio-composites and bio-based materials [1] in almost every industry as the world becomes more conscious of its impact on the environment. Mycelial materials have the potential to become as accessible and in demand as plastics without the harmful effects on wildlife and the environment. However, the possible applications of mycelial materials has not been fully explored. The introduction of mycelial materials started in the packaging and distribution industry as a suggested replacement for polystyrene foam [1] but with further research on the physical properties of mycelial bio foam it could be used in many other fields.

There exist a variety of bio-based materials, some of which include Sorghum, wood and paper [2], but mycelium is especially interesting because of its flexibility and malleability. There is not much literature on the mechanical behavior of mycelium composites. The existing literature does not focus on the properties of mycelial materials and therefore allows for only limited cross examinations to evaluate possible uses for this bio composite. In this paper, we study a mycelial mold's deformation in different scenarios. The objective to obtain more information on the properties of the mycelium composite. We evaluated a method to make mycelial sam-

¹Danielle Brennan is with the Department of Chemical Engineering, Howard University, Washington, DC 20059, USA danielle.brennan@bison.howard.edu

²Keene Chin is a PhD student with the Soft Machines Lab, Carnegie Mellon University Robotics Institute, Pittsburgh, PA 15213, USA keenec@andrew.cmu.edu

³Dr. Carmel Majidi is with the Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA cmajidi@andrew.cmu.edu



Fig. 1. Typical Structure of Mycelium-based foam [1]

ples. We test the samples through mechanical and online simulations and performed related calculations.

II. RELATED WORK

This section is a brief overview of previous applications and suggested uses for mycelium composite.

A. Packaging Applications

Fossil-based plastics are used many different sectors due to the structural integrity it provides and its bending abilities. For mycelium-based materials to "replace" plastics they must have comparable properties. The first literature on mycelium-based materials were focused on the low-density and good insulation properties of the material [1] making it suitable for packaging material and a replacement for polystyrene [3].

B. Design and Architectural Applications

The initiative to find renewable and biodegradable materials has extended to the construction industry as there is a constant need for housing with our growing population [4]. Mycelium is of interest because of its low cost, density and energy consumption. There is still more research to be done before this bio-composite can be fully implemented, but there are still companies that manufacture mycelium-based furniture and other structures. Water absorbency and compressive strength of the bio-composite are also important factors in determining the efficacy of mycelial material in the design and architecture field [5].

C. Suggested Uses

Efforts have been made to see the electronic application this but more research has to be done before it is implemented [6]. Mycelium has also been introduced to the fashion industry through "mycelium leather", a sustainable latticework of mycelium. Other industries are slowly looking to mycelium as their material for sustainable transformation.

III. METHODOLOGY

This project used two different methods to create the mycelial samples: the flour method and direct method [7]-[9].

A. Direct Method

With this method three different substrates were used to observe the mechanical differences between the mycelial materials formed: sawdust, wood chips and rye berries.

- The *Pleurotus ostreatus* (*P. ostreatus*) culture was introduced to the sterilized substrates utilising the techniques used to prepare grain spawn [7].
- The mycelium material was transferred to a lined, sterilized cardboard mold and allowed to sit for 4-5 days out of direct sunlight [8].
- The mycelial samples were allowed to sit in a larger container for 2 days to form a white outer layer.
- The sample was then baked at 200 degrees Fahrenheit for 30 minutes.
- The mycelial samples are now ready for use.



Fig. 2. Grain Spawn with Rye Berry Substrate

B. Flour Method

- Rye berries were used as the substrate for the grain spawn along with the *Pleurotus ostreatus* (*P. ostreatus*) fungus culture [10].
- The mycelium was allowed to grow on this substrate for 7 days then activated the mycelium with a mixture of flour and water and let sit in a warm area for 4-5 days [9].

- After the mycelium is activated a few teaspoons of flour is mixed in and the mixture is put into a sterile mold and was allowed to sit out of direct sunlight for 5-6 days.
- The sample air dried for 2 days and then was baked at 200 degrees Fahrenheit for 30 minutes.
- The mycelial sample is now ready for use.



Fig. 3. Mycelium in Substrate using Flour Method

After the samples are made a MATLAB script analyzed the deflection of the sample during the experiment to determine its Elastic Modulus. The MATLAB frame by frame analysis script created to analyze the footage of the mycelium cantilever test was tested using footage of a cantilever test with a sample of expanded polystyrene foam. The script's calculated Elastic modulus was within the range of values provided in the literature. The calculated Elastic Modulus was 2.888 MPa and the range is 1.379 MPa to 3.309 MPa.

IV. EXPERIMENTS

The molds are tested to obtain values for physical properties to compare with existing values [1]. The bio-composite undergoes two mechanical tests: The cantilever test and the living hinge test.

A. The Cantilever Test

There are various setups for the Cantilever Test, for this research we use the Cantilever Beam with a concentrated load at any point. The deflection of the material being tested, v and the deflection at the end of the beam, δ_B are determined by the following equations:

$$v = \frac{-Px^2}{6EI} (3L - x) \quad (1)$$

$$\delta_B = \frac{PL^3}{3EI} \quad (2)$$

Where P is the force acting at the point along the length of the sample, L , x is a point along the span of the sample. E

represents the modulus of elasticity and I is the area moment of inertia of the sample's cross section.

We obtain the deflection from the mechanical tests, after using an image processing script on Matlab² to analyze video footage of the sample experiencing varied forces. The equations above would then be rearranged in order to find the elastic modulus of the mycelial sample.

B. Living Hinge Test

The Living Hinge Test is a simple mechanical test to determine at the angle at which the material will break, hence allowing us to examine the flexibility and durability of the material. The living hinge sample is formed by attaching a circular shaped piece of polymer clay to the mold that the mycelial material will grow in. This creates a rectangular shape with a "bridge" in the center, where the material will be bent.

V. RESULTS

After the 7 day growth period, the rye berry sample became white with mycelium while the wood chip and sawdust samples showed no sign of growth.



Fig. 4. Mycelial Bio-foam with rye-berry substrate

Based on the Elastic modulus range [11] of mycelium provided in existing literature, we were able to hypothesize the relationship between the deflection of the material and the elastic modulus with and without added weight.

The flour method also proved to be unsuccessful when the sample became contaminated. We postulate that insect eggs or larvae were in the flour, incidents like these may be avoided by sterilizing the flour by applying heat.

A. Testing

The sample's texture was not very conducive for the living hinge test and as a result by visual inspection the sample broke at about a 35 degree angle.

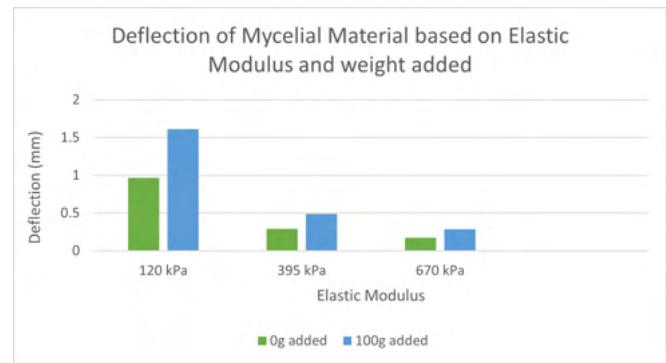


Fig. 5. Graph showing relationship between Deflection and Elastic Modulus [11]

The MATLAB script could not analyze the footage of the mycelium sample being tested as the first weight (data point) caused the sample to break.



Fig. 6. Cantilever Test Setup with Mycelium Sample



Fig. 7. Mycelium Sample at Maximum Deflection when Weight is added

VI. DISCUSSION

We expected that the mycelial sample would be able to withstand much more weight than it did. The point at which the sample broke there was an existing fracture, this was as a result of the mycelium not fully developing around the substrate particles. Many factors could have contributed to the not ideal mycelium growth.



Fig. 8. Mycelium Sample after breaking

A. Factors that could have affected the health of the Mycelium

The health of the Mycelium may have been affected by the surrounding temperatures, altitude, air quality and humidity. In an ideal environment all these variables would have been kept constant and adjusted for optimal mycelium growth, but those conditions could not have been met in our testing environment.

B. Factors that could affect the Mycelium's ability to connect parts of the mold

Mycelial materials are formed by the mycelium growing and binding its substrates' particles together to form one solid bio-composite. Our mycelial sample did not cohere as expected and this could be caused by using insufficient amounts of fungus culture, compression applied to the sample and the saturation level of each substrate particle in the culture.

VII. CONCLUSION

Throughout this project, we observed the behavior of our mycelium material and the process by which the bio-composite is made. For optimal mycelial samples we would have a controlled environment and introduce more fungus culture to the suitable substrate. After creating the mycelial bio foam we would conduct a cantilever test and use our MATLAB script to analyze the video footage and determine its Elastic Modulus. Following the collection of this data we would conduct a Finite Element Analysis to test possible mechanism designs. There are still numerous possible uses for mycelial materials yet to be discovered. With much more research and ideal conditions we believe mycelial materials could be used to make compliant mechanisms.

ACKNOWLEDGMENT

This work was supported by the CMU Robotics Institute Summer Scholars (RISS) Program, and special thanks to Rachel Burcin and Dr. John Dolan for facilitating the program this year virtually and providing tremendous support. I would like to thank my mentors Dr. Carmel Majidi and Keene Chin from the Soft Machines Lab for their assistance with this project.

REFERENCES

- [1] C. Girometta, A. M. Picco, R. M. Baiguera, D. Dondi, S. Babbini, M. Cartabia, M. Pellegrini, and E. Savino, "Physio-mechanical and thermodynamic properties of mycelium-based biocomposites: A review," *Sustainability*, vol. 11, no. 1, 2019. [Online]. Available: <https://www.mdpi.com/2071-1050/11/1/281>
- [2] M. A. Curran, *BIOBASED MATERIALS*. Hoboken, NJ: John Wiley & Sons, Inc., 2010, pp. 1–19.
- [3] J. Jose, K. N. Uvais, T. S. Sreenadh, A. V. Deepak, and c. R. Rejesh, "Investigations into the development of a mycelium biocomposite to substitute polystyrene in packaging applications," *Arabian Journal for Science and Engineering*, vol. 46, 2021. [Online]. Available: <https://doi.org/10.1007/s13369-020-05247-2>
- [4] M. Jones, A. Mautner, S. Luenco, A. Bismarck, and S. John, "Engineered mycelium composite construction materials from fungal biorefineries: A critical review," *Materials & Design*, vol. 187, no. 108397, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0264127519308354>
- [5] N. Attias, O. Danai, T. Abitbol, E. Tarazi, N. Ezov, I. Pereman, and Y. Grobman, "Mycelium bio-composites in industrial design and architecture: Comparative review and experimental analysis," *Journal of Cleaner Production*, vol. 246, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0959652619339071>
- [6] E. S. L. Vasquez and K. Vega, "From plastic to biomaterials: Prototyping diy electronics with mycelium," 2019. [Online]. Available: <https://doi.org/10.1145/3341162.3343808>
- [7] F. Mushrooms, "The easy way to make mushroom grain spawn for growing mushrooms at home," *YouTube*, Aug. 2020. [Online]. Available: <https://www.youtube.com/watch?v=nQ1uDfsT4yI>
- [8] E. Design, "How to make a cardboard growth form," *YouTube*, Aug. 2018. [Online]. Available: <https://youtu.be/A51zw66oI80>
- [9] Cecasil, "Mycelium molds."
- [10] M. Haneef, L. Ceseracciu, and C. et al. Canale, "Advanced materials from fungal mycelium: Fabrication and tuning of physical properties," 2017. [Online]. Available: <https://doi.org/10.1038/srep41292>
- [11] M. R. Islam, G. Tudry, R. Bucinell, L. Schadler, and R. C. Picu, "Mechanical behavior of mycelium-based particulate composites," *Journal of Materials Science*, vol. 53, 2018. [Online]. Available: <https://link.springer.com/article/10.1007/s10853-018-2797-z>

Automatic Detection of Road Work and Construction with Deep Learning Model and a Novel Dataset

Brian Chen¹, Robert Tamburo², Srinivasa Narasimhan²

Abstract— Automating the process of identifying and understanding road work and construction zones is significant and beneficial for the safety of road workers, drivers and obtaining traffic conditions in real-time. While tasks like object detection and segmentation have been well studied in the field of computer vision and achieve promising results, there has not been any work focusing on the reasoning of construction zones. Despite plenty of transportation related images and video data of available online, there is no dataset specifically created for construction zones. In this paper, we introduce a new dataset collected from both online and the real world. We discuss how we use a deep learning framework to automatically and efficiently label the dataset from rich and diverse sources. The results are promising and show that our dataset can be used to automatically detect real-world construction zones. Moreover, we conduct experiments and analysis to better understand the features that are crucial in identifying construction zones and found that color features are important. Finally, in an effort to improve detection accuracy we experiment with using segmentation masks of different objects as additional input for identifying construction zones.

Index Terms— Computer Vision for Transportation; Recognition

I. INTRODUCTION

Object detection and classification techniques based on deep learning models have been applied to real-world applications in different aspects, especially in the transportation field. For example, intersection cameras provide real-time traffic status updates using car detection and object segmentation. Autonomous cars rely on sensors, cameras, and machine learning to understand the environment in order to determine the appropriate control actions, which requires computer vision techniques, both in 2D and 3D. Due to the complex nature of real-world environment, it is difficult to be reliable and timely in the prediction of the deep learning models, since artificial intelligence methods are unable to achieve a comprehensive understanding of the surrounding environment comparable to human beings. Because of this limitation, even the state-of-the-art methods may cause serious safety issues if the environment scene is wrongly interpreted. In this work, we focus on enhancing our knowledge in the construction zone, as it will affect the models' output significantly and relatively low attention has been drawn to this specific area.

¹Chen is with the University of Illinois at Urbana-Champaign. He completed this work during his internship at CMU. brian5@illinois.edu

¹Narasimhan and Tamburo are with the Robotics Institute, Carnegie Mellon University. srinivas@andrew.cmu.edu, rtamburo@cmu.edu



Fig. 1. This figure demonstrates various forms of construction zone from different sources in our construction zone dataset. The top left image is obtain using Google engine. The top right image is taken from Taipei. The bottom two images are collected from Pittsburgh. Construction zone has diverse appearances, depending on the location, the type of construction, and various other factors.

Construction zone is inevitable on the road, and serious accidents are more likely to happen in nearby area due to the presence of workers on the road and unexpected changing road conditions. For example, distracted drivers bring less attention to unexpected road conditions and hazards. Moreover, understanding construction zone allows to send real-time updates to pedestrians, pedal cyclists, and drivers about traffic status and construction progress. However, as current deep learning methods rely heavily on large-scale datasets to train models, the lack of construction zone specific dataset prevents us from applying state-of-the-art methods to gain a better understanding of construction zone. Existing datasets either only concentrate on detecting a small subset of objects found in construction zones, such as traffic signs [1]–[4], or focus on a more general view of traffic and congestion [5]–[10], ignoring the details of the construction zone, which plays a crucial part of transportation applications.

Therefore, in this work we aim to address these issues and conduct an analysis of construction zone. We present a novel construction zone labeled dataset and explore the usefulness of this dataset in real world by using our trained models in different scenarios. Currently this dataset supports binary label - construction and non-construction. Images are collected from diverse sources so that the model trained from it can be generalized to unseen environment in the future. Moreover, we conduct analysis to understand what features are critical for classification. We learned that the color is

a crucial factor for models to classify construction zone by conducting experiments and feature map visualization. Additionally, we dive into different approaches such as utilizing pseudo label and segmentation masks to expedite the process of dataset collection and improve model performance.

Our main contributions include:

- 1) create a construction dataset consisting of 4429 images, images were scraped from google image searches and captured from smart phones in Pittsburgh and Taipei
- 2) train a construction identification model and utilize it to generate pseudo label to allow the model to automatically annotate images
- 3) conduct analysis in the effects of color and segmentation masks for identifying construction zone

II. RELATED WORK

In this section, we review the datasets related to transportation and construction zone. Specifically, we will go over some traffic sign datasets, general scene datasets, and driving/transportation datasets.

Traffic signs are common components in construction zone. As recognition and detection for traffic signs have been studied in many previous works, some of the presented datasets are solely applicable for traffic signs. For example, Lisa Traffic Sign Dataset [1] is a US traffic signs dataset, which includes the original video tracks of all annotated images/frames of traffic signs. German Traffic Sign Recognition Benchmark (GTSRB) and German Traffic Sign Detection Benchmark (GTSDB) [2], [3] are traffics sign datasets collected in German for similar purposes. The former dataset is for traffic sign detection while the latter is for traffic sign classification, with more than 50,000 images. Benchmark Tsinghua-Tencent 100K [4] contains images with traffics signs taken under various illumination and weather conditions. Class label, bounding box, and segmentation mask are available in the dataset. Despite the fact that these traffic sign datasets provide useful information for construction zone understanding, these are not out-of-box datasets that could facilitate the construction zone identification.

There are several nature object and scene datasets, such as Pascal Visual Object Classes (VOC) [5], Microsoft COCO [6], ADE20K [7], which are more designed for general purpose scene classification, segmentation, and detection. In the context of driving datasets, KITTI [8], MIT DriveSeg datset [9], and BDD100K [10] are well-developed datasets, with high-quality pixel level segmentation, high resolution and frame rate, and other features for driving applications. BDD100K has multiple tasks such as lane marking and drivable area detection, which are relevant to the construction zone. Besides real-world datasets, several works utilize computer graphics to generate synthetic datasets, such as Virtual KITTI dataset [11] and SYNTHIA dataset [12]. Moreover, there are also datasets specifically designed for persons and pedestrians detection, *e.g.* CityPersons [13], EuroCity Persons [14]. While these datasets include many objects relevant to construction zone, such as fence and individual standing wall, which makes it sufficient for most

common cases, they lack particular details and information of construction zone and its related objects, such as the type of construction zone, etc. In our work, we propose to construct a rich dataset specifically for construction zone analysis.

III. METHOD

We first go over the definition of our construction zone dataset. We then move on to the methodology and strategies used for collecting dataset. Moreover, we explain our choices of models for identifying construction zones. We then describe how we try to utilize the trained model to increase the efficiency of labeling process with pseudo label. Finally, we discuss the procedure of making use of segmentation masks as additional input signals.

A. Construction Zone Dataset

Definition. The most important aspect of creating a dataset is to determine what we expect from the dataset and what information should be included. We define the construction zone in our dataset as road construction, sidewalk construction, bike-lane construction, and any other construction that would directly effect normal, daily transportation of regular uses. In other words, we do not collect images of pure construction zone, *e.g.*, an area that is not on the road or does not affect the traffic, or building construction site. Additionally, portraits of construction-related objects are ignored as it does not provide much information about the condition of transportation. We accept images with arbitrary aspect ratios. Images taken from the human-level height and intersection camera are desired, while viewpoints from extreme angles are not considered, *i.e.* satellite-level or ground-level viewpoints. The dataset should cover a wide range of diversity, including but not limited to the location, weather, time of days, environment, type of roads, configuration of roads, type of constructions, and the type of barriers.

Collect with Google engine. To minimize the labor cost of annotations for our dataset, we exploit the power of Google search engine. Google engine is beneficial because we can inherently obtain the labels of the images by specifying the query keywords. To automate this process, we write a python script that can process multiple keywords together and download images accordingly. At this point, the main problem becomes what the query keywords should be. To ensure that the dataset is sufficiently diverse and general to be applied to the real world scenarios, we design a complex set of query keywords in the following format

$$[\text{multiple objectives}] + \text{noun}$$

Originally, the objectives we used include weather condition, location, and others. While the nouns of interest include "road construction", "road maintenance", and "road work", eventually we only use "road construction" since the other two usually return duplicate images as the first one. Yet, we conclude that Google search engine cannot returning relevant images while the compound keywords are over-complex. Specifically, when the keyword has one noun and more than two objectives, the returned images mostly only satisfy one

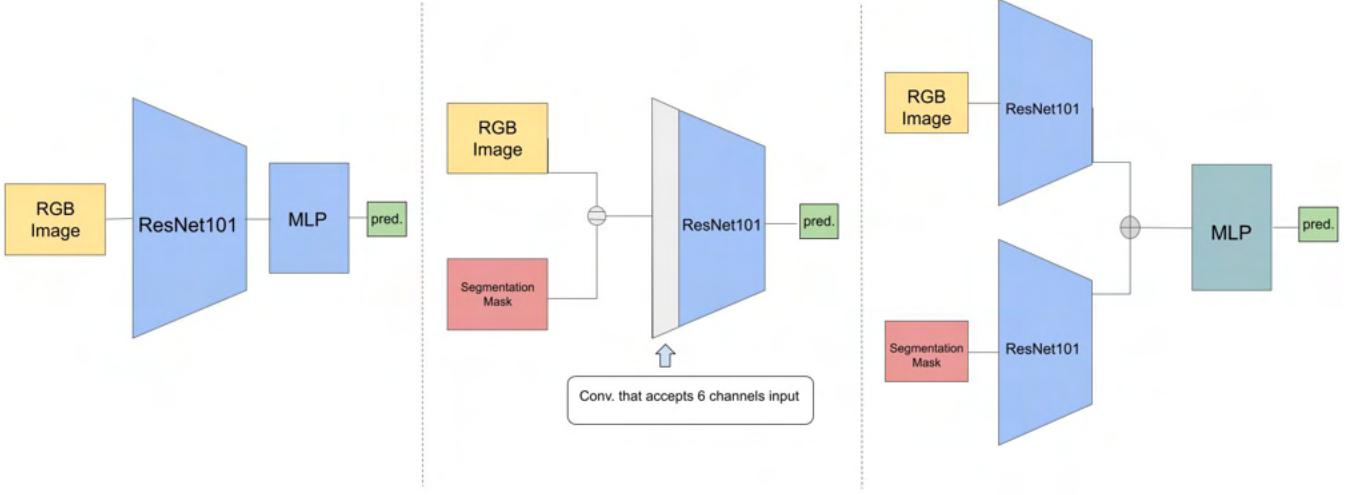


Fig. 2. This figure illustrates the pipeline used for different experiments. The left is for the basic construction zone classification using ResNet101. The middle and right are the two architectures we use when having segmentation masks as additional input. In the middle pipeline, we only changes the first layer of ResNet101; the RGB image and segmentation masks are stacked and fed into the model. In the right, we use 2 branches pipeline in order to learn better features.

or a few query keywords. The most dangerous case is that the image only satisfy the objectives, instead of actually containing the construction zone. Therefore, even though using multiple keywords can provide richer annotations, using only one objective and one noun for the query keywords actually yield more effective results for our purposes.

Filter unsuitable images. Although simple keywords facilitate Google engine to return higher quality images, the returned images are still sometimes unsuitable for a dataset. For example, it may return images with different types and formats, such as synthetic images or slides of a presentation due to the high functionality of Google engine. Even though we explicitly prohibit Google engine from returning images other than photos, using the built-in filter function, the problem still remains. Furthermore, we may still get images that are not construction zone while the query keyword is simple and contains the keyword "construction". Other issues include getting images with bad scale, watermarks, and from unacceptable viewpoints. Therefore, to collect a reliable dataset, manual labeling and checking is still a mandatory procedure with the use of Google engine in our experiments. Google engine can only serve as a convenient tools to gather images efficiently.

Diverse data sources. To verify the model trained on the dataset G collected from Google engine can be generalized to real world, we need other datasets from different environments or distributions. For this purpose, we collect dataset by taking pictures in the cities where we live in. Due to remote working, we are fortunate to be able to collect photos from two cities across the ocean - Pittsburgh and Taipei, as shown in Fig. 1. This greatly increases the diversity of the dataset and allows us to justify the generalization of G . Since we collect dataset by ourselves, we can label it more

efficiently by only considering certain types of photos in a period. To improve the performance of the dataset, we try to collect images in nearby area of construction zone so that the distribution of construction and non-construction images are more balanced. This forces the models to focus on the key features of the construction zone for identifying, instead of relying on other noise features. In addition, we collect images during driving. We deploy cameras at different locations on the car and set up a remote shutter control so that we could safely and efficiently take photos while driving.

B. Model Architecture

For simplicity, the classification model for identifying construction zone is using ResNet101 [15] as the backbone, followed by multi-layer perceptron (MLP), as shown in the left of Fig 2. ResNet101 is a popular backbone for many larger networks in the field of computer vision. It features the residual blocks, which allow building larger architectures by minimizing the gradient vanishing issue and preserving the features and information in previous layers. In addition, we apply an existing segmentation model - HRNetV2 [16] - to obtain the segmentation masks for our dataset. The segmentation masks are then fed into the ResNet101 backbone with the original RGB image to provide more information about the geometric structure of the image.

To accommodate the additional input to the model, we adopt two modifications to the original architecture, as illustrated in Fig. 2. First, the number of the input channel to the first layer of ResNet101, which is a convolution layer, is changed from three to six. By expanding the input channels, we can stack the original RGB image and the segmentation mask, and use it as the input to the model to enhance the inherent information. The second modification is to have two branches of ResNet101 model for learning different features

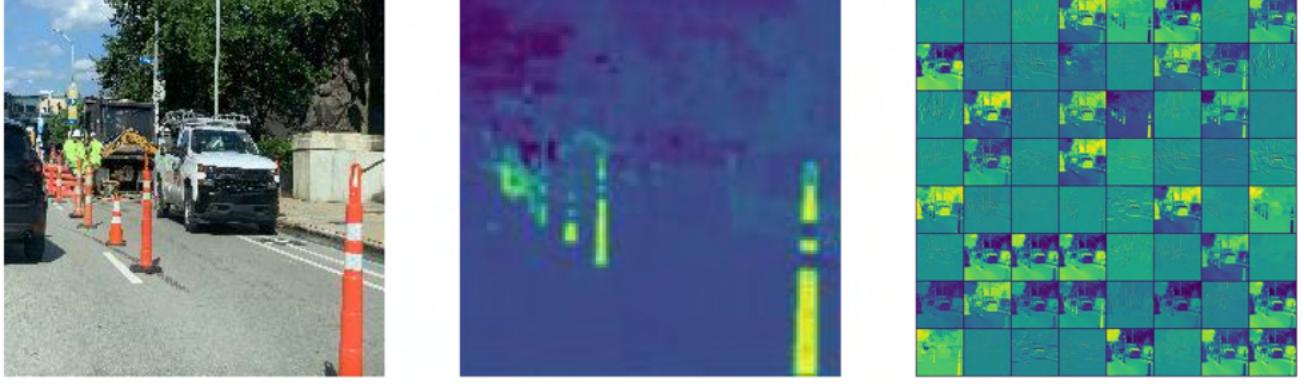


Fig. 3. This figure shows that color plays an important role in identifying construction zone. The left is the input of the model. The middle is a selected feature map of the first convolution layer of ResNet101, which shows the model relies on the orange color to make decision. The right is all the 64 feature maps of the first convolution layer of ResNet101.

TABLE I
THIS TABLE SHOWS THE RESULTS OF THE MODEL TRAINED BY USING
THE DATASET G .

Test Dataset	Accuracy	Number of Images
A	0.8801	273
B	0.9150	153
C	0.6849	948
D	0.7184	1612
E	0.6453	113

for RGB images and segmentation masks respectively. The two embedded features are then concatenated and fed into a MLP to generate the final prediction of our model.

In addition, we allow the trained model to label unseen images, when the model is confident in its prediction. The trained model is considered confident when the difference in the output logits of the top-2 prediction is greater than 0.15. With this automatic annotation function, we utilize the information of an annotated dataset to obtain a new labeled dataset annotated by our models. We call these labels pseudo labels since they are annotated by the trained models, in contrast to the ones annotated by humans. Finally, we use this dataset with pseudo labels to update our trained model again.

To further improve the models and interpret what the model has learned to identify construction zone, we design experiments to investigate the behavior of the model. Specifically, we visualize the feature maps of the ResNet101 model and analyze the results. Furthermore, we run the model with gray scale image input to demonstrate the importance of the colors in the input image.

IV. RESULTS

A. Datasets

Datasets. We test the model on multiple datasets we built as described in Section III-A. Since the entire construction dataset is collected over time at different locations throughout the project, we divide the dataset into multiple small subdatasets. We will refer subdataset as dataset in the rest of the article.

The first dataset G is built using google search engine and requires filtering. Before filtering out unsuitable images, there are about 3.6K images. After manually filtering, the total size of the dataset is 1.2K, where 64% of them are construction images and the rest are non-construction images. About 66% of the returned images from Google engine are not suitable for our dataset.

We have other 4 datasets collected in Pittsburgh, and 1 dataset collected in Taipei. Dataset A and B are collected while walking in Pittsburgh. Dataset C and D are collected while driving in Pittsburgh. Dataset E is collected while riding scooter in Taipei.

Evaluation. The accuracy is defined by computing the number of correct predictions divided by the number of total predictions.

To evaluate how well the model trained on dataset G can be generalized to the real world, we train the model using dataset G and tested on all other datasets individually. Table I summarizes the model performance on various datasets. Accuracy on dataset C , D , E is relatively low compared to performance on dataset A and B . It is conjectured that this is due to the inconsistency in annotations. For example, one might annotate images as construction zone based on whether there exist traffic cones and construction fences on roads. On the other hand, one might consider images as construction zone only when there is an active construction zone. The second reason might be the changes in dataset domains, which requires additional model adaptation to fix the issue. Additionally, we run an experiment by combining all datasets into one and randomly split them into 70% for



Fig. 4. This figure shows the prediction by model trained on dataset **G**. The predicted labels are shown on the left. The red border indicates that the images are wrongly classified.

training and 30% for evaluation. We achieve 88% on this setup. Therefore, we expect the model trained by our dataset to have comparable capability of identifying construction zone in real world.

For consistency, we complete the rest of the experiments using dataset **G** as the training set and dataset **A** as the test set.

B. Feature Exploration

We visualize the feature maps of the first 3 layers of the ResNet101 to understand what the model learns. Fig. 3 shows that except the horizontal and vertical edges, the orange region of the feature map is highlighted in a single feature map. This indicates that color features are captured by the model while identifying construction zones. Furthermore, the input signals are turned into a single channel to see how the model performs. In other words, we transfer RGB images into gray scale images. Specifically, we use the PyTorch built-in function to map RGB images to gray scale and set all the RGB channels be the gray-scale value so that we do not need to modify the original model architecture. Table II shows that model with only the gray scale images perform significantly worse than the RGB images. We can infer that the model cannot well identify the construction zone solely using the geometric properties of the images. This is consistent with our claim that the color is a crucial factor for identifying the construction zone.

In Fig. 4, we show several images from dataset **C** and the corresponding predictions from our model. The two images in the top-left are wrongly classified as construction zone. We suspect that this is because of the objects with bright red color that appear in the image, such as the building wall and red car. Interestingly, the third image in bottom row, which also has large area of red color, is correctly classified. Probably this attributes to the fact that our model also relies on the geometric structure of the images to make decision. In the bottom-left, two images of highway road works are

TABLE II
THIS TABLE SHOWS THE EFFECT OF COLORS AND SUGGESTS THAT MODELS HEAVILY RELY ON COLOR TO IDENTIFY CONSTRUCTION ZONE.

Input	Accuracy
RGB image	0.8801
Gray image	0.7109

wrongly classified, it is conjectured that this is due to the insufficient diversity of our dataset.

C. Pseudo Label

The pseudo label experiments are conducted using the dataset **G** and **A**. Specifically, we split the dataset **G** into approximately 40% for initial training. The remaining 60% are labeled by the trained model. As mentioned in Section IV-A, we evaluate the model on dataset **A**. The confident threshold for the model is set to 0.15. We observe that the model's performance drops by 11% after being trained on the images with pseudo labels. There are two factors that could potentially lead to this result. First, the hyperparameter - confident threshold - might not be optimized. There is a trade-off between the number of images being labeled and the precision of the pseudo labels. If the model is forced to annotate images only when highly confident, the number of annotated images will be small, and vice versa. Currently it is not clear what threshold value is optimal. Secondly, the model might not be able to correctly annotate images since the provided training dataset is not huge enough. The lack of training images is a classic problem in computer vision. It is planned to conduct more experiments with the pseudo label setup when more images are available to resolve this issue.

TABLE III

THIS TABLE SHOWS THE EFFECTS OF DIFFERENT SEGMENTATION MASKS.

Filters	Accuracy
w/o filter (150) categories	0.7875
All relevant (21) categories	0.7985
Most relevant (7) categories	0.7985
Road only	0.7839

D. Segmentation Masks

We try two different model architectures to accommodate the additional segmentation mask input, as described in III-B. The first architecture does not perform well compared to the one without using any segmentation masks, which contradicts to our hypothesis that the segmentation masks are useful. The first possible reason is that the segmentation mask from HRNetV2 is not good enough to serve as the input. The second explanation is that the model is not strong enough to learn and extract the information in the segmentation masks. As we only do a simple trick to ResNet101, we change the architecture by making a 2 branches architecture, where each branch is a duplicate ResNet101 model, as shown in the right of Fig. 2. We find that the accuracy increases by 12% by using the 2 branches architecture.

Segmentation masks are obtained by using the HRNetV2, which is pretrained on the ADK20 dataset. Since this dataset has 150 categories, we filter out the categories that are unlikely to appear in our construction dataset to reduce the noise in the segmentation masks. Since it is hard to determine which categories should be included, we apply different sets of mask filters to see what the optimal set of categories is. Specifically, we try using (a) all the categories in the original dataset, (b) all the relevant categories that might appear in our dataset, (c) the essential categories that are very likely to be helpful for identifying the construction zone, and (d) only the "road" category. Table III shows a summary of the experiments. We conclude that filtering does help improve the performance by a margin. However, as the performance is still worse than the baseline, a better model needs to be considered.

V. DISCUSSION

We demonstrate that the model trained from our construction dataset is capable of identifying construction zone in several real-world scenarios. We also discover that our current method cannot utilize the segmentation masks as additional inputs to improve the performance of our models. It is expected that better framework for training models and more labeled data are required for the segmentation experiments.

In future work, we will expand our dataset to include more annotations of objects commonly found in construction zones. The annotations include the bounding box of chosen objects, the type of construction zone, the blocking condition of lanes, etc. Moreover, we would like to review the definition of the construction zone, to ensure the consistency and robustness of our dataset. For example, we will specify the portion of construction zone on an image that is required to label that image as construction zone. In addition, we would like to run and test the model on a larger dataset, such as images from bus cameras and intersection cameras. In terms of model functionalities, we will integrate our model with other methods that could detect and segment construction objects.

ACKNOWLEDGMENT

This work is supported by CMU Robotics Institute Summer Scholars (RISS) and sponsored by NSF REU under Grant No. ECCS2038612. Special thanks to the RISS program organizers Rachel Burcin and Dr. John Dolan.

REFERENCES

- [1] A. Mogelmose, M. M. Trivedi, and T. B. Moeslund, "Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1484–1497, 2012.
- [2] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Networks*, vol. 32, pp. 323–332, 2012, selected Papers from IJCNN 2011. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0893608012000457>
- [3] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark," in *International Joint Conference on Neural Networks*, no. 1288, 2013.
- [4] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- [5] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, pp. 303–338, 06 2010.
- [6] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015.
- [7] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5122–5130.
- [8] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.
- [9] L. Ding, J. Terwilliger, R. Sherony, B. Reimer, and L. Fridman, "Mit driveseg (manual) dataset," 2020. [Online]. Available: <https://dx.doi.org/10.21227/mmke-dv03>
- [10] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," 2020.
- [11] Y. Cabon, N. Murray, and M. Humenberger, "Virtual kitti 2," 2020.
- [12] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. Lopez, "The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes," 2016.
- [13] S. Zhang, R. Benenson, and B. Schiele, "Citypersons: A diverse dataset for pedestrian detection," 2017.
- [14] M. Braun, S. Krebs, F. Flohr, and D. M. Gavrila, "Eurocity persons: A novel benchmark for person detection in traffic scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, p. 1844–1861, Aug 2019. [Online]. Available: <http://dx.doi.org/10.1109/TPAMI.2019.2897684>

- [15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” 2015.
- [16] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, “Deep high-resolution representation learning for visual recognition,” 2020.

Joint SLAM on Multiple Monocular Cameras for Legged Robots

Thomas Detlefsen¹, Sayan Mondal², and Matthew Travers³

Abstract—Legged robots have a distinct advantage over other platforms, such as wheeled or tracked robots, because of their ability to adapt to highly-variable terrain. However, the ability to dynamically adapt to changing terrains comes at a cost; on board sensors such as inertial measurement units, cameras, LIDAR, etc. are often violently shaken, producing signals that are hard to interpret or outright fail altogether. To this end, it has been shown that when the camera source is placed on a legged platform, visual-inertial simultaneous localization and mapping (SLAM) methods experience degraded performance due to unpredictable sensor motion [1]. Even robust SLAM systems that can recover from some unpredictability in sensor measurements tend to suffer from degraded performance due to the abrupt and motions and vibrations on sensors that corrupt measurements. To address this, we introduce a method for performing association between sensor measurements on a rapidly moving camera and increasing re-association confidence through the use of recurrent neural networks (RNNs) and a learning-based tracking algorithm.

Index Terms—Monocular Vision, Features, Re-identification, Simultaneous Localization and Mapping (SLAM)

I. INTRODUCTION

Legged robots are crucial to robust applications of robotics where other platforms fail. They have the ability to navigate natural terrain such as ones with uneven or soft surfaces, whereas other platforms require a continuous support surface to move effectively. [2] The capability to maneuver across different terrain makes legged robots a common choice for applications such as industrial inspection, exploration, or search and rescue missions. [3]

In recent years, SLAM algorithms have vastly improved, but they still perform poorly when applied to systems with rapid and dynamic motion. [4] Systems that rely heavily on visual information often incur a lower cost to implement, but do not perform well on legged platforms because of the unpredictable motion of a legged robot. These systems experience increased effectiveness at lower speeds where movement is more controlled, but they are still not as effective as on other platforms [1] and provide a limitation for the speed of the platform when it is physically able to move faster.

In this work, we introduce a design for a robust visual-inertial SLAM algorithm that identifies visual features of the environment and re-associates them to the map within multiple monocular camera frames on board a legged robot.

¹Thomas Detlefsen is with the Electrical Engineering Department at the University of Pittsburgh, Pittsburgh PA 15213, USA ted32@pitt.edu

²Sayan Mondal is with the Biomedical Engineering Department at the University of Pittsburgh, Pittsburgh PA 15213, USA sayanmon@andrew.cmu.edu

³Matthew Travers is with the Robotics Institute at Carnegie Mellon University, Pittsburgh PA 15213, USA mtravers@andrew.cmu.edu

Our method is split into four modules: detection, classification, tracking, and re-identification. The detection module recognizes environment features and records their position relative to the robot. The information about these features is then used to classify the features with increasing confidence using the classification module. These features and classifications inform the tracking module which generates a unique identifier for each feature so that it can be re-identified between time-steps and between camera views. The association module uses the feature identities and attributes to re-identify features to map the environment and localize the robot.

Figure 1 shows a simplified example of this system in action. In the first time-step, features are extracted from the image taken by the camera and unique identifications are generated for each feature. In the next time-step, the previously detected features (shown in red) are re-identified and new features are detected and fit to the environment. Again, in the third time-step, the previous features (shown in orange) are re-identified by the system.

The method is implemented and demonstrated by representing visual features as objects on a conveyor belt moving between two cameras. This representation allows us to abstract away the details of feature selection in SLAM, as this ability has been demonstrated consistently in many modern SLAM systems. [1], [5]-[7] This abstraction allows us to focus on the performance of the tracking and re-identification of features in a controlled environment.

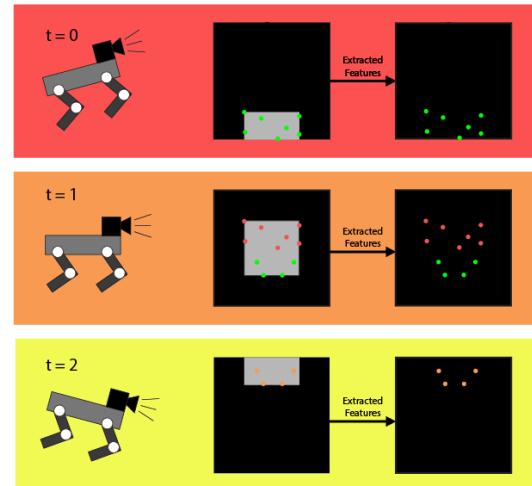


Fig. 1. A simplified example of the system is shown above. In the first time-step, features are identified and unique identifications are generated. In the next time step, the previously detected features (shown in red) are re-identified and new features are added to the map. In the final time-step previously detected features (shown in orange) are again re-identified.

II. RELATED WORK

Within the field of localization for mobile robotics, there are a large variety of SLAM algorithms suited for different purposes. Some of these systems are designed for general use on mobile robots but have degraded performance when adapting to specific use cases, such as legged perception. This section discusses state-of-the-art general SLAM algorithms and those more specifically suited for SLAM on legged robots.

A. State-of-the-art general SLAM Algorithms

This section provides a brief overview of some state-of-the-art SLAM algorithms suited for general use in mobile robotics. These methods perform well when there is less frame-to-frame movement, but as the amount of displacement between camera measurements increases, the performance of general use systems degrades.

1) *ORB-SLAM2*: ORB-SLAM2 is a feature-based SLAM system designed for monocular, stereo, and RGB-D cameras. ORB-SLAM2 is the continuation of ORB-SLAM [5], a monocular vision SLAM system that uses the same features throughout tracking, mapping, re-localization, and loop closure. ORB-SLAM2 demonstrates an increase in accuracy with the use of bundle adjustment over iterative closest points used in previous methods. This system features a real-time camera re-localization method that allows the algorithm to recover from failure, but it still has difficulty with consistent camera movement. [6]

2) *VINS-Mono*: VINS-Mono is a visual-inertial SLAM system that utilizes a monocular camera and inertial measurement unit (IMU) measurements. The primary contribution of VINS-Mono is the ability to initialize the system from an unknown state which was previously difficult due to the IMU. In previous visual-inertial systems, the IMU was required to start from an unknown moving position to calibrate the camera and IMU together. This method outperforms ORB-SLAM because of the incorporation of IMU measurements which constrain the pose estimation of the mobile robot. This inclusion of IMU measurements is still not sufficient to accurately perform SLAM on a dynamic system such as a legged robot. [7]

B. State-of-the-art SLAM Algorithms for Legged Perception

Recently, there have been developments in the field of robust SLAM systems for legged perception. These methods often utilize graph optimizations and introduce further constraints than general methods to restrict the graph if too few features are found in common between camera measurements. While these systems improve over general use SLAM systems, they still fail to perform at higher speed movement where more instability is introduced to the cameras of the robot.

1) *PUT SLAM*: PUT SLAM is a system reliant on RGB-D sensors that is specifically designed for the application of urban search and rescue missions on legged robots. PUT SLAM uses bundle adjustment to create a map of features optimize them with the sensor trajectory. The graph created is

constructed of features and sensor poses. While PUT SLAM performs better than conventional state-of-the-art SLAM systems, it still is not robust to legged robots with rapid and unstable gaits. [1]

III. METHOD

This section discusses the details of our robust monocular camera SLAM system for legged systems. To abstract away the details of feature selection, as this ability has been demonstrated consistently in many modern SLAM systems, our method represents features as objects moving between multiple cameras on a conveyor belt. Because of this, we employ four modules to our system: detection, classification, tracking, and association. An overview of this method is shown in figure 2.

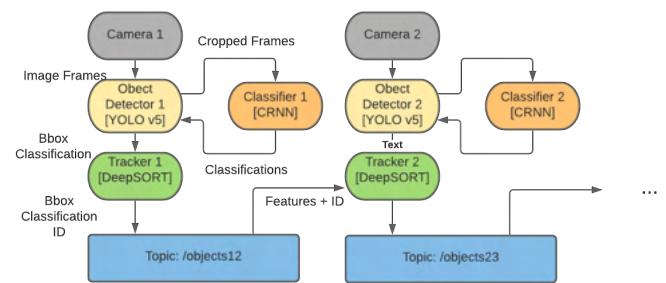


Fig. 2. The system diagram for our SLAM system is shown above. Images are passed from the camera to the detector where each detected object is classified and returned. These classifications and bounding boxes are passed to the tracker where unique object IDs are generated and the information is provided to the next system so that it can inform detections.

A. Detection

For the initial detection of objects, this method uses the YOLO v5s network, which is the most recent iteration of [8]. YOLO uses a single CNN to extract features from an entire image and maps the detections onto a grid. Then the confidence score is created and the bounding boxes are generated. YOLO v5 has a pre-built classifier, but in this work, the YOLO v5s classifier is substituted for an RNN based classifier, because the original classifier does not take into account temporal information from previous detections. The advantage of YOLO v5s in this context is that its speed is suitable for real-time detection at high frame rates allowing the system to gather more information on the objects to make accurate future predictions.

B. Classification

The classification network uses a joint CNN-RNN network (known as CRNN), depicted in figure 3. The cropped images from the object detector are fed into the network at each time step, and processed by the CNN, the extracted features are fed to the RNN which uses the hidden states of the previous time step to create a better estimation to classify the correct object. At each time step, the classifier outputs a probability distribution that describes the likelihood of the classification of the object. Over time, as more views

of the object are available, the classification of the object improves. This classification is returned to YOLO v5s where the bounding box and classifications are associated.

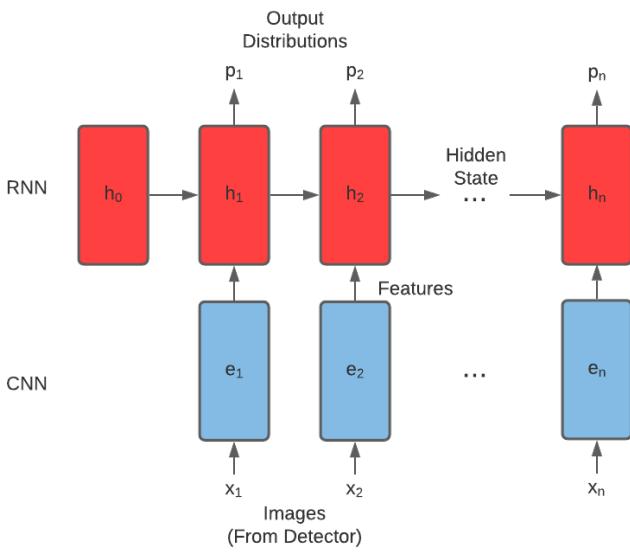


Fig. 3. A visualization of an unrolled CRNN is shown above. At each time step, an image is fed into a CNN and the features are passed to the next state in the RNN. The RNN uses the hidden states from the previous time step to generate a probability distribution across the classes based on features and previous classifications.

1) Training: For training, we have generated a dataset of over 200 videos of objects moving across a conveyor belt and sorted the data for classification. To train the CRNN, we use a sliding window with length n that includes $n-1$ images of an empty conveyor belt and one image of the object. In the following video index, the window slides, decreasing the number of conveyor belt images and increasing the object images until all of the images from the video are used. This allows for uncertainty when an object first enters the camera view, then increasing confidence as more images of the object are found.

C. Tracking

For the tracking and re-identification of objects, this system employs the DeepSORT network. DeepSORT is an extension of SORT which takes into account visual features and object trajectory to perform multi-object tracking. [9] This system receives the bounding box and classifications from YOLO v5s and generates an ID based on the features and previous states of the object. The ID and classification are then made available for other tracking modules for the other camera systems to access and perform re-identification.

IV. RESULTS

To evaluate the SLAM system proposed in this paper, we represent features as objects moving past a camera on a conveyor belt. This method allowed us to abstract away the details of feature selection in SLAM, as this ability has been demonstrated consistently in many modern SLAM systems. While we were have not yet achieved full integration across

multiple monocular cameras, we were able to integrate the detection and tracking modules and compare the classification of the CRNN to that of YOLO v5s.

A. Detection and Tracking Integration

For initial testing of the system proposed in this paper, only the tracking and detection models were integrated. This means that objects were detected and classified by YOLO v5s and the bounding boxes and classifications were passed to DeepSORT where IDs were generated for each object so that they could be tracked throughout the view of the camera.

During this testing, there were a fixed number of objects from each class sent down the conveyor belt in quick succession. We used the tracker to keep a running total of the number of objects from each class and compared this to the actual number of objects sent (one example from a test is shown in figure 4). We found that while objects consistently maintained the same ID, the objects were somewhat frequently classified incorrectly.

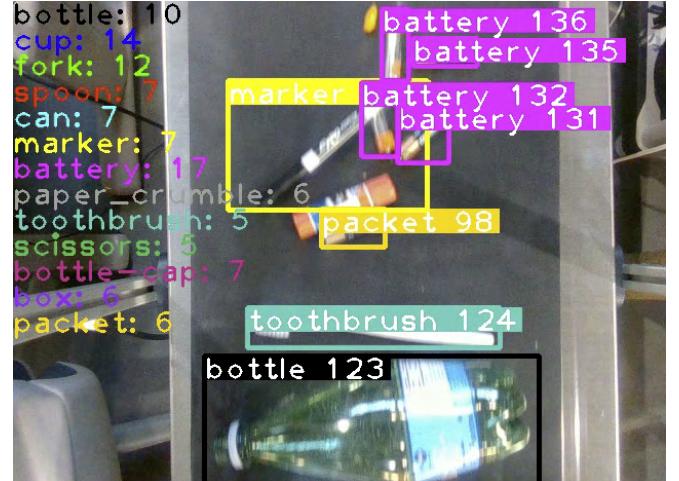


Fig. 4. The figure shown above is an example of the results from the integration of the detection and tracking modules. On the left side of the screen, there is a counter which lists the number of each class that has passed through the camera frame. As can be seen, some objects are miss-classified, and some are not classified at all.

The frequent classification changes and miss-classifications show that there is a need for a more robust classification module. Specifically, one with temporal knowledge can increase the confidence of new classifications based on knowledge of previous states. For this reason, the CRNN classification module is effective for this purpose.

B. Classification

The CRNN classification module was tested and compared to the classification native to YOLO v5s. This comparison was made by sending one object at a time down the conveyor belt so that only one object was visible to the camera at once. The results of this experiment are shown in figure 5. The chosen object (a bottle) entered the view of the camera after 5 frames. As seen in figure 5, the classification from YOLO v5s is extremely sporadic for the first 8 frames, and

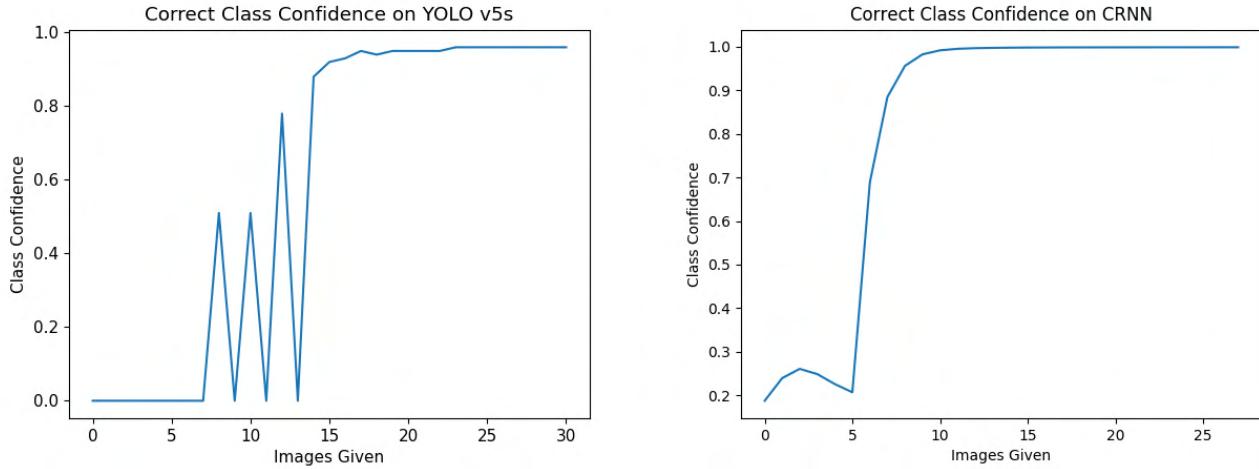


Fig. 5. The figure shown above demonstrates the ability of the CRNN to perform more consistent classification than time-independent networks like YOLO v5s by using the previous states to inform present classification. An object appears starting in frame 5. YOLO v5s does not classify the object correctly for a few frames and is inconsistent in its classification for the first few frames. The CRNN classifies the object correctly immediately then the confidence increases consistently until it is almost 1.

eventually gains confidence. In contrast, the classification from the CRNN is more smooth and the classification rises consistently, as it uses information from the previous states to inform the next classification.

V. DISCUSSION AND FUTURE WORKS

This work demonstrates the design of a joint monocular SLAM system robust to rapid camera motion due to the movement of a legged body. The results shown in the previous section demonstrate the current progress of this research. This section will discuss how these results build a path for the SLAM system described and what the pursuit of this research will be moving forward.

The first example in the results, showing the integration of the tracking and integration system shows the ability of the system to detect and track objects frame to frame while maintaining the same identity. This will be extremely important for identifying and tracking features with rapid body movement. The second set of results helps to show how including information from previous classifications helps to improve future classifications, giving them higher confidence after fewer frames. Both of these results show the initial phases of implementing a robust SLAM algorithm for legged robots.

In future iterations of this project, we plan to integrate the classification module into the monocular-camera network so that object classification confidence can be observed increasing as the object travels across the view of the camera. Once this is complete, we plan to extend this to the multiple monocular camera system originally proposed so that objects can be re-identified outside of the original camera view. This implementation would ideally be in real-time as YOLO v5s was chosen specifically for its speed and accuracy results. Once this implementation is completed, we plan to adapt this multi-camera monocular SLAM system to a legged robot

where the method can be evaluated and tweaked for the greatest success.

ACKNOWLEDGEMENT

This paper was made with the support of the National Science Foundation under REU Grant 1659774. Thomas Detlefsen would like to thank Ms. Rachel Burcin, Dr. John Dolan, and the rest of the Carnegie Mellon University Robotics Institute Summer Scholars Program Staff for providing a fantastic research opportunity. Thomas Detlefsen would also like to thank Dr. Matthew Travers, Sayan Mondal, and the rest of the biorobotics lab for their support and guidance.

REFERENCES

- [1] M. Nowicki, D. Belter, A. Kostusiak, P. Cížek, J. Faigl, and P. Skrzypczyński, “An experimental study on feature-based slam for multi-legged robots with rgb-d sensors,” *Industrial Robot: An International Journal*, vol. 44, pp. 428–441, June 2019.
- [2] J. Machado and M. Silva, “An overview of legged robots.”
- [3] D. Bellicoso, M. Bjelonic, L. Wellhausen, K. Holtmann, F. Günther, M. Tranzatto, P. Fankaußer, and M. Hutter, “Advances in real-world applications for legged robots,” *Journal of Field Robotics*, vol. 35, pp. 1311–1326, November 2018.
- [4] C. Cadena, L. Carlone, H. Carrillo, and Y. Latif, “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age,” *IEEE Transactions on Robotics*, vol. 32, pp. 1309–1332, 2016.
- [5] R. Mur-Artal, J. Montiel, and J. Tardos, “Orb-slam: a versatile and accurate monocular slam system,” *IEEE Transactions on Robotics*, vol. 31, pp. 1147–1163, 2015.
- [6] R. Mur-Artal and J. Tardos, “Orb-slam2: an open-source slam system for monocular, stereo, and rgb-d cameras,” *IEEE Transactions on Robotics*, vol. 33, pp. 1255–1262, 2017.
- [7] T. Qin, P. Li, and S. Shen, “Vins-mono: A robust and versatile monocular visual-inertial state estimator,” *IEEE Transactions on Robotics*, vol. 34, no. 4, 2018.
- [8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” 2016.
- [9] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” 2017.
- [10] M. Ramezani, G. Tinchev, E. Iuganov, and M. Fallon, “Online lidar-slam for legged robots with robust registration and deep-learned loop closure,” pp. 4158–4164.

Weakly Supervised Classification of Vital Sign Alerts as Real or Artifact

Arnab Dey^{1,2}, Mononito Goswami¹, Joo Heung Yoon³, Gilles Clermont³, Michael Pinsky³, Marilyn Hravnak³, Artur Dubrawski¹

Abstract—Numerous studies have shown that a significant proportion of clinical alarms are false. This often leads to alarm fatigue in clinical personnel, inevitably compromising patient safety. To combat this issue, researchers have attempted to build Machine Learning (ML) models capable of accurately adjudicating Vital Sign (VS) alerts raised at the bedside of hemodynamically monitored patients as real or artifact. Previous studies utilize supervised ML techniques, which require substantial amounts of hand-labeled data to train reliable classifiers. However, in our observation, harvesting manually annotated ground-truth labels on data can be costly, time-consuming, and mundane, and is one of the key limiting factors that prevent widespread adoption of beneficial ML capabilities in healthcare. As an alternative, we explore the use of multiple imperfect heuristics to automatically assign probabilistic labels to unlabeled training data using data programming methodology. These heuristics provide weak supervisory signals to train accurate and efficient downstream classifiers, sidestepping the demanding requirements of previously used approach. The primary goal of our study is to substantiate the efficacy of weakly supervised classification of respiratory rate and oxygen saturation alerts occurring in data from monitored intensive care patients. We found Random Forest (RF) models trained on probabilistic labels generated using the proposed approach to perform comparably or outperform similar models trained on ground truth labels, highlighting the potential of data programming as an efficient and practical alternative to supervised learning in healthcare applications of ML.

Index Terms—Machine Learning, Weak Supervision, Vital Sign Alerts

I. INTRODUCTION

Intensive care patients who are at risk of cardiorespiratory instability (CRI) undergo continuous monitoring of vital sign (VS) parameters such as electrocardiography, plethysmography, pulse oximetry, and impedance pneumography. Recent advances in commercial bedside monitoring devices have made the sustained tracking of the physical state and health of a connected patient a real possibility. Without these devices it is practically impossible for medical practitioners to continually and attentively observe fast-evolving and heterogeneous VS parameters. However, even modern commercial devices have surprisingly inadequate support for identifying abnormal physiological variables in the form of simple exceedances of pre-determined normality thresholds

¹Arnab Dey studies at the Georgia Institute of Technology, Atlanta, GA, USA adey43@gatech.edu

²Mononito Goswami and Artur Dubrawski are with the Auton Lab, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA mgoswami@andrew.cmu.edu, awd@cs.cmu.edu

¹ Joo Heung Yoon, Gilles Clermont, Michael Pinsky, and Marilyn Hravnak are with the University of Pittsburgh School of Medicine, Pittsburgh, PA, USA yoonjh@upmc.edu, {cler, pinsky, mhra}@pitt.edu

[1]. However, it is not uncommon for patients to have atypical VS parameters due to occasional movement, electrical interference, or loose sensors [2].

Indeed, numerous studies have shown a large percentage of these VS alerts to be false, or more formally, artifact [3] - either of mechanical, electrical, or physiological nature [2], [4]. Additionally, medical practitioners may be exposed to up to 1,000 alarms per Intensive Care Unit (ICU) shift [5]. The sheer amount of alarms in tandem with the high rate of artifacts can quickly lead to alarm desensitization and burnout in healthcare professionals. Multiple studies have concluded that the resulting alarm fatigue can have severe negative consequences for patient safety with several incidents resulting in preventable harm or even death of a subject [3], [5], [6]. In addition to the added stress placed on medical practitioners, the frequent alerts from patients can also lead to increased physiological stress, metabolic impairment, sleep disturbance and even death [6]. The U.S. Food and Drug Administration (FDA) has reported over 500 alarm-related patient deaths in the short span of five years [5].

Previous attempts to combat alarm fatigue have relied on advancements in adaptive filtering or explored the use of various Machine Learning (ML) paradigms, namely supervised, semi-supervised and active learning. However, all these methodologies require varying quantities of manually and pointillistically labelled data. Labeling alerts as real or artifact is often not only time-intensive, but also laborious, expensive, and mundane tasks that pull experienced clinicians away from their patients. Furthermore, traditional ML paradigms do not easily adapt to evolving clinical expertise and changing problem definitions due to their reliance on pointillistically annotated data which must be re-labeled to accommodate each such problem redefinition.

As an alternative, data programming methodology proposes to harvest general heuristics that clinicians would use to label the data by hand, and use them collectively to probabilistically reconstruct the labels in even vast amounts of unlabeled reference data. The hope is that downstream models trained with such automatically annotated data would perform as well as would the models trained on data labeled in a data-point-by-data-point fashion, while of course the human effort needed to accomplish that can be vastly reduced.

Recent work suggests that the proposed *weak supervision* (WS) methodology can indeed accomplish such goals in some healthcare time-series applications [7], [8]. In this paper, we demonstrate the potential utility of this framework to adjudicate bedside alerts as real vs. artifact in high-density waveform VS data collected in intensive care settings.

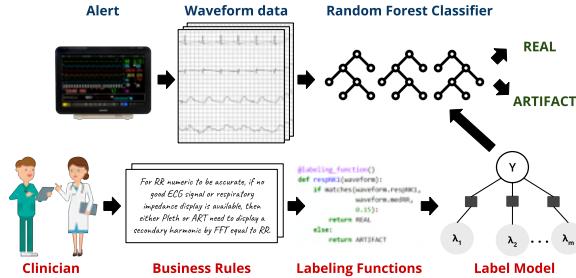


Fig. 1. Shown is a diagram outlining the Weak supervision pipeline for the binary classification of vital sign alerts. Heuristics given by domain experts are encoded into labeling functions whose votes are fed into a generative model. This generative model then outputs probabilistic labels that are used in tandem with feature sets generated from vital sign time-series data for training a downstream Random Forest Classifier.

II. RELATED WORK

A. Alarm Fatigue

Alarm fatigue caused by high rates of artifact VS alerts is a widely-studied problem and a variety techniques have been adopted to combat it in previous research. Most approaches fall into two main categories: (1) artifact reduction, and (2) artifact detection. The former approach attempts to reduce the number of artifact alerts produced through internal improvements within the vital sign monitors and other biosignal measuring devices. Advancements in adaptive filtering and other techniques to reduce artifacts in real time within the monitor itself have been developed [9]–[14]. But, due to the wide frequency range diverse nature and causes of artifact alerts, [15], [16], the problem of alarm fatigue still persists [2]. Our paper instead aims to tackle alarm fatigue and the high rates of artifact alarms through the latter approach, which focuses on post-measurement artifact detection and classification.

B. Clinical Settings

A large amount of research has been conducted on post-measurement artifact detection in the past, but most either look at ambulatory settings or in the context of wearable devices and smartphones – settings which are fundamentally different from VS alert classification in the clinical settings due to differences in physiological state of users, data quality, rates of motion and noise artifact, amount and type of available data, a priori likelihood of artifact, and primary differences in the types of artifact that is detected [17]–[19]. While in a clinical setting, a VS alert is classified as real or artifact, some aforementioned papers classify biosignals themselves as real or artifact.

C. Machine Learning Paradigms

Prior research on artifact detection strictly in the clinical setting has been conducted, but most papers combat alarm fatigue through the use of traditional ML pipelines such as fully-supervised (FS), active, semi-supervised, and federated learning [2], [20], [21]. These papers have made great strides

in VS alert classification and in fact, the medical data curated by [critical care alert group] used in the paper by Chen et al. for VS alert classification was also used in this work [2]. However, these previously studied learning paradigms require vast amounts of initial training data to train efficient and accurate classifiers, which is greatly inconvenient, and even impractical in certain situations. There is a distinct lack of analysis on classifiers trained in data and label sparse environments using pipelines expressly suited for this situation. Moreover, to the best of our knowledge, our work is the first to apply weak supervision to the problem of VS alert classification.

D. Weak Supervision to Solve Clinical Problems

Prior work on applying weak supervision to solve a gamut of clinical problems ranges from detecting aortic malformation using cardiac MRI sequences [22], detecting abnormal heartbeats [8], to detecting seizures using EEG data [7]. Our work follows [8] in being among the first to apply weak supervision to time series data using heuristics defined directly on time series, compared to prior work such as [7] which used heuristics defined on auxiliary modalities such as text.

III. METHODOLOGY

A. Problem Formulation

Broadly, given an alert, our goal is to classify it as a real or artifact. We specify each alert as a 4-tuple $\mathcal{A}_i = (\text{pid}, \tau, t, d)$, where aid is a unique alert ID, $\tau \in \{\text{RR}, \text{SpO}_2\}$ is the alert type, t and d are the starting time and duration of the i^{th} alert. We assume that each alert is associated with an unobserved true class label $\mathcal{Y}_i^* \in \{0, 1\}$, where 0 denotes real and 1 denotes an artifact; and that for the duration of the alert, we have access to time series data \mathcal{T}_i which includes both waveforms such as ecg leads II and III and numerics such as HR, potentially sampled at different frequencies. We aim to use clinical intuition and expert knowledge encoded in several noisy heuristics to obtain labels to train an downstream classification model \mathcal{M} . We define each heuristic, alternatively called a labeling function (LF), denoted by $\lambda : \mathcal{T} \times \mathcal{A} \rightarrow \{-1, 0, 1\}$ directly on timeseries data. A LF either abstains (-1) or votes for a particular class (0, 1) given an alert \mathcal{A} and its associated waveform data \mathcal{T} . While we do not expect LFs to have perfect accuracy or recall, we do expect them to have better than random accuracy where they vote. Starting with n patient alerts $\mathcal{X} = \{(\mathcal{A}_i, \mathcal{T}_i)\}_{1 \dots n}$ and m labeling functions $\Lambda = \{\lambda_i\}_{i=1 \dots m}$, our goal is to learn a label model \mathcal{L} which assigns a probabilistic label $\hat{p}(y | \Lambda)$, $y \in \{0, 1\}$ to each patient alert in X .

The label model learns from the overlaps, conflicts and optionally dependencies between the LFs using a factor graph as shown in Fig. 1. In this work, we assume the LFs to be independent given the true class label. While this assumption may not always stand, most prior work [7], [8], [22] has shown that the label model works well in practice.

We use the label model proposed in [23] to learn the joint distribution between the unobserved

B. Vital Sign Data

Our data was collected at one of the University of Pittsburgh Medical Center hospitals, and curated and de-identified at the University of Pittsburgh whose Institutional Review Board deemed this research did not qualify as human subjects research. Cardiorespiratory vital sign alert data consisting of a variety of waveforms and numerics was collected by a Philips IntelliVue MX800 Monitor, from a mix of Intensive Care Unit (ICU) and Step Down Unit (SDU) patients. Numerics, including Respiratory Rate (RR), Heart Rate (HR), oxygen saturation (SpO_2), and telemetric oxygen saturation (SpO_2T) were sampled at 1 Hz. Waveform data including lead II and lead III Electrocardiograms (ECG), plethysmographs (pleth), telemetric plethysmographs (plethT), arterial pressure waveforms (ART) derived from an indwelling arterial catheter, and respiratory waveforms (resp) from impedance pneumography were all sampled at various frequencies. ECG lead II and lead III were sampled at both 250 Hz and 500 Hz. Pleth, plethT, and ART were all sampled at 125 Hz, and the resp waveform was sampled at 62.5 Hz. It is also worth noting that the featurizations done on the ART data were removed later in the study when it was discovered that a large portion of the data was missing or incomplete for both RR and $\text{SpO}_2/\text{SpO}_2\text{T}$ alerts.

C. Vital Sign Alert Events

We determined both RR and $\text{SpO}_2/\text{SpO}_2\text{T}$ vital sign alerts by analyzing the RR numeric and $\text{SpO}_2/\text{SpO}_2\text{T}$ numeric, respectively, on 4 factors: (1) *duration* - at least 5 minutes of the respective numeric data was present, (2) *persistence* - at least 70% of the numeric values exceeded respective thresholds (< 10 breaths per minute or > 29 breaths per minute for RR and < 90% for $\text{SpO}_2/\text{SpO}_2\text{T}$), (3) *tolerance* of 5 minutes suggesting that consecutive alerts < 5 minutes apart were combined, and (4) density expectation of 65% of numeric values present at a 1 Hz sampling frequency. These factors ensured that the VS alerts we analyzed contained continuous spaced anomalies with minimal interruption and were sufficiently long to have clinical relevance. Similar to prior work by [2], [20] we only used the first 3 minutes of each alert event for both RR and $\text{SpO}_2/\text{SpO}_2\text{T}$ alert classification. However, we broke each 3 minute alert window into three 1 minute windows, primarily as a way to artificially boost the sample size. The ground truth label for each of the alert windows was assumed to be the same as the ground truth for the parent event. In the rest of this paper, RR or $\text{SpO}_2/\text{SpO}_2\text{T}$ alerts refer to these 1 minute alert event windows.

D. Expert Knowledge Informing VS Alert Classification

Manually classifying artifact VS alerts is an arduous, repetitive, yet sufficiently objective process, largely governed by a set of guiding principles or "business rules" based on visual distinction and clinical intuition [20]. Most business

```
@labeling_function()
def respNK1(waveform):
    if matches(waveform.respNK1, waveform.medRR, 0.15):
        return REAL
    else:
        return ARTIFACT
```

Fig. 2. This sample RR LF demonstrates the general design of labeling functions. Heuristics suggested by domain experts can be easily encoded as a set of simple conditional statements. In this specific case, when the value for the median RR derived from respiratory waveform data is within 15% of the median RR numeric, the VS alert is classified as real.

Alert	LF Name	Polarity	Coverage	Overlaps	Conflicts	Emp. Acc.
RR	respNK1	[0,1]	0.844	0.844	0.458	0.803
	respNK2	[0,1]	0.844	0.844	0.458	0.519
	respINT	[0,1]	0.809	0.809	0.448	0.735
	respFFT	[0,1]	0.844	0.844	0.458	0.847
	respHeight	[0]	0.234	0.234	0.062	0.860
	plethFFT	[1]	0.228	0.228	0.183	0.958
	plethNK1	[1]	0.305	0.305	0.216	0.937
	plethNK2	[1]	0.264	0.264	0.222	0.971
SpO_2	plethNK1	[0,1]	0.657	0.657	0.192	0.877
	plethNK2	[0,1]	0.657	0.657	0.192	0.853
	plethINT	[0,1]	0.657	0.657	0.192	0.896
	plethFFT	[0,1]	0.657	0.657	0.192	0.939
	pulsatility	[0]	0.125	0.113	0.111	0.677
	pulsatilityT	[0]	0.095	0.022	0.022	0.957
	plethMatchesECGiinNK1	[0,1]	0.044	0.044	0.042	0.682
	plethMatchesECGiinNK2	[0,1]	0.050	0.050	0.048	0.640
	plethMatchesECGiinINT	[0,1]	0.040	0.040	0.038	0.700
	plethMatchesECGiifFT	[0,1]	0.050	0.050	0.048	0.800
tachypnea		[1]	0.450	0.389	0.121	0.762

TABLE I

SUMMARY STATISTICS FOR ALL LFS. POLARITY LISTS THE CLASSES A LF VOTES FOR, WHERE 1 CORRESPONDS TO A **REAL** ALERT AND 0 DENOTES AN **ARTIFACT**. IN OUR WORK, WE USED A MIX OF UNI AND BIDIRECTIONAL LFS. ADDITIONALLY, THE HIGH OVERLAP AND CONFLICT WITH THE GENERALLY GOOD COVERAGE INDICATES THAT THE LFS WERE WELL-SELECTED. THE EMPIRICAL ACCURACY OF EACH HEURISTIC IS PROVIDED AS TOOL FOR ENHANCING THE READER'S UNDERSTANDING OF THE WEAK SUPERVISION PIPELINE, BUT GENERALLY THIS STATISTIC WOULD BE UNAVAILABLE IN PRACTICE, SINCE THE CALCULATION RELIES ON GROUND TRUTH LABELS.

rules are based on the apparent disagreement between observed numerics, and corresponding numerics derived from recorded waveform data. For instance, most business rules to distinguish between real and artifact RR alerts are based on discrepancy of observed RR and the numeric derived from the resp, pleth, plethT, and ART waveforms. In this study, however, we were unable to derive RR plethT and ART waveforms after finding a large portion of the data for these waveforms to be missing or incomplete. Similarly, $\text{SpO}_2/\text{SpO}_2\text{T}$ alerts are more likely to be artifacts when the observed HR does not match HR derived from the pleth/plethT waveform. Our label model leveraged the overlaps and conflicts between labeling functions built on different core methodologies to probabilistically label training data.

Some other business rules compared the HR derived from ECG lead iii to that computed from the pleth waveform, and examined whether patients are experiencing tachypnea (rapid breathing, with $\text{RR} > 20$) during an oxygen saturation alert. To improve reliability, some business rules also checked whether resp and pleth waveforms were too low or displaying a lack of pulsatility.

E. From Expert Knowledge to Labeling Functions

Since most business rules relied on RR and HR numerics derived from the recorded waveforms, we developed multiple core methodologies with wide ranging accuracy, to compute these numerics. For most business rules relying on derivations of RR and HR, it was important to be able to compute the primary/secondary harmonics and locate peaks in different waveforms. For instance, the RR closely corresponds to the median number of peaks and the primary harmonic of a clean resp waveform. We compute the former using a modified version of SciPy’s peak detection algorithm [24] and an extrema extraction algorithm proposed in [25] as implemented in Neurokit2 [26]. We computed the primary harmonic of the resp waveform by locating the highest peak of a periodogram modified by the Bohman windowing function. Prior to using the resp waveform, we processed it subjecting it through linear detrending followed by a fifth order 2Hz low-pass IIR Butterworth filter [25].

To derive RR from the pleth and ART waveforms, we first processed them via a different, novel, multi-step methodology, which involved interpolating the tips of the peaks found using SciPy’s peak detection algorithm via spline interpolation. This was done to ‘clean’ the pleth and ART waveforms which experienced significant noise, making RR estimation using our core methodologies hard. For SpO₂/SpO₂T alerts, we derived the HR numeric from ECG lead ii and iii using the same core methodologies, after employing an ECG cleaning technique proposed in Neurokit [26].

Finally, we translated our business rules into labeling functions building on the aforementioned core methodologies. As an example, Figure 2 illustrates one such LF, comparing the observed median RR (waveform.medRR) with the RR derived from resp using the methods proposed in [25] and implemented in Neurokit (waveform.respNK1). We implemented a total of 8 and 11 noisy heuristics for the binary classification of RR alerts and SpO₂/SpO₂T alerts, respectively. Table I presents summary statistics for all RR and SpO₂/SpO₂T LFs.

F. From Labeling Functions to Alert Classifier

We trained the label model \mathcal{L} defined in the previous section using LFs for respective VS alerts, to obtain probabilistic labels for our training data. We used label model implementation in Snorkel [27] for the same. Samples not covered by any LF were filtered out, and the remaining probabilistic labels produced by the label model were translated into crisp binary training labels, which were then used to train a Random Forest (RFs) model [28] to classify VS alerts as real versus artifact. RFs have been widely used in literature to learn complex decision boundaries for various classification problems [29], [30] and have also been shown to be effective for learning discriminative models of real versus artifact VS alert classification [2]. We trained RF models with 1000 decision trees having a maximum depth of 5 implemented using scikit-learn [31].

IV. EXPERIMENTAL SETUP

A. Featurization

In order to train the RF models, we not only used the features computed for use by our LFs such as the wave height of the resp waveform (respHeight), RR derived from a modified periodogram of the resp waveform (respFFT), etc., but also extracted features from the raw waveforms and numerics themselves by computing a set of aggregate statistics (mean, standard deviation, kurtosis, skewness, median, 1st and 3rd quartile). For the RR alerts, we subsequently dropped the features calculated from the ART, plethT, and ECG lead iii waveforms, and the SpO₂T numeric, due to more than 75% of the alerts not having the prerequisite waveform or numeric data required to compute such features. For SpO₂/SpO₂T alerts, we only dropped features calculated from the ART waveform, for the same reason. The complete set of featurizations for both alert types can be found in the appendix. Next, we replaced any missing values remaining in the data after incomplete features were removed with either a 0 or -1 depending on the normal range of the features.

B. Baselines and Evaluation

We compared our weakly supervised RF model with its fully supervised counterpart trained using ground truth labels (Fully Sup.), probabilistic labels produced by the label model (Prob. Labels), and RF models trained using majority vote (Majority Vote) instead using the data programming label model. The majority vote label model predicted labels simple based on what the majority of LFs voted. All models were trained on a leave-one-patient-out (LOPO) cross-validation setting, where the training data comprised of data from all but one patient, and later were tested on the held-out patient. This setting ensures that the models do not inadvertently fit to patient specific characteristics to artificially inflate their performance.

We compared all the models based on a number of performance metrics such as precision, recall, F1, accuracy and AUC. We also computed metrics of practical utility such as the false positive rate at 50% true positive rate (FPR 50% TPR), true positive rate at 1% FPR (TPR 1% FPR) etc. These metrics were computed across all LOPO cross validation folds.

All models and labeling functions were implemented using Python programming language (version 3.8.1), and experiments carried out on a computing cluster with 64 CPUs equipped with AMD Opteron 6380 processors having a total of 252 GB RAM.

C. Additional Research Questions

In addition to examining the efficacy of weakly supervised models for VS alert classification, we aimed to answer the following research questions from our experiments.

1) *What patterns are our RF models learning?:* Interpretability is important when ML models are deployed in clinical settings, especially when using black box models such RFs. To this end, we used *Gini importance* (GI) [28]

and *permutation feature importance* (PFI) [32] to determine features which our weakly supervised model relied on the most while making predictions. Since GI can be inflated for high-cardinality features, PFI was also analyzed to reliably understand trained RF models, in line with prior work conducted in different settings [30]. GI and PFI was evaluated by accessing the feature importance attribute of a trained scikit-learn RF classifier, and using the `permutation_feature_importance` function in scikit-learn, respectively.

2) *How useful is the waveform data?:* Because previous work on VS alert classification, by [2] for example, did not have access to waveform data, we became curious about the predictive utility of waveform data for classifying VS alerts. To that end, we conducted ablation experiments by withholding waveform features while training and validating our weakly and fully supervised models using the same LOPO cross-validation procedure. However, we must note that the LFs informing the weakly supervised RF still had access to requisite waveform data, and therefore these experiments are not completely indicative of settings with a lack of waveform data.

3) *Does the weakly supervised model predict telemetric and non-telemetric SpO_2 alerts equally well?:* Since SpO_2 and SpO_2T alerts were mixed during the main experiments, the difference in classifier performance between telemetric and non-telemetric oxygen saturation alerts was briefly explored.

V. RESULTS

A. Performance Metrics

For the RR alerts, the various performance metrics shown in Tables IV & V highlight the weakly supervised pipeline's surprising, but superior performance over the supervised learning paradigm. Analysis we conducted on the performance metrics for the oxygen saturation alerts yielded more expected results, with the supervised learning pipeline performing slightly better than the weak supervision pipeline. However, considering the immense advantage of accessing ground truth labels for training, the weak supervision pipeline's comparable performance to the supervised learning pipeline was still impressive. Tables IV & V also indicate that all the RR pipelines performed better than the $\text{SpO}_2/\text{SpO}_2\text{T}$ pipelines. This finding is consistent with prior work by [2]. However, the performance gap we found between the two alert types was slimmer, likely due to the inclusion of waveform data in our work.

B. ROC-AUC

Based on the log-scale AUC plots for RR alerts shown in Figure 3 (i & ii), it is clear to see that the weak supervision pipeline has a higher TPR and TNR at nearly every FPR and FNR, respectively. For $\text{SpO}_2/\text{SpO}_2\text{T}$ (Plots iii & iv in Figure 3), although the weakly supervised pipeline is not better, it performs comparably to the fully supervised pipeline, despite not having access to ground truth labels, underscoring

the impressive capabilities of the weakly supervised Data Programming learning paradigm.

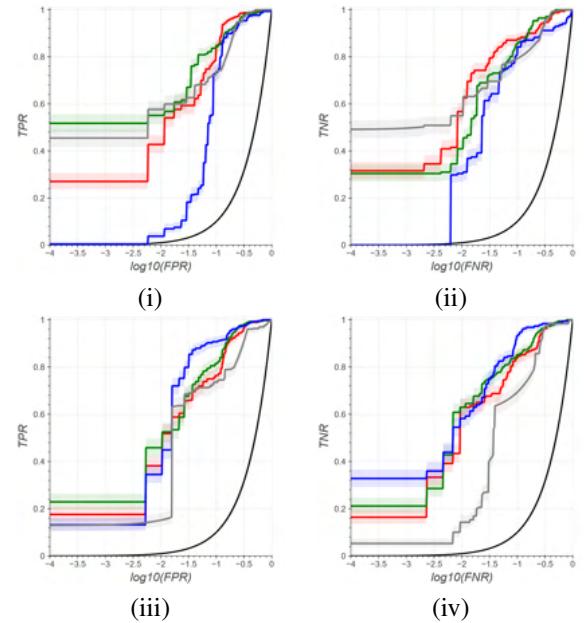


Fig. 3. A set of 4 log-scale ROC-AUC plots for RR & $\text{SpO}_2/\text{SpO}_2\text{T}$ alert classification are shown above, each with the weak supervision (red), majority labeler (green), supervised learning (blue), and probability labels (grey) pipelines. The first two plots (i & ii) correspond to RR Alert classification, while the last two (iii & iv) correspond to $\text{SpO}_2/\text{SpO}_2\text{T}$ Alert classification. The ROC-AUC plots highlight the WS pipeline's ability to keep up with the fully supervised pipeline for the $\text{SpO}_2/\text{SpO}_2\text{T}$ alerts, and even beat it for the RR alerts.

C. Answers to Additional Research Questions

1) *Our RF models for WS and FS pipelines are learning similar patterns:* Despite discrepancies in GI and PFI, overlap in features such as the standard deviation of the resp waveform (`std_resp`) for RR alerts, and HR derived from the pleth waveform using a modified periodogram (`plethFFT`) for the $\text{SpO}_2/\text{SpO}_2\text{T}$ alerts indicate that both the weakly and fully supervised RF models may be learning similar patterns.

2) *Waveform data is helpful for RR alerts, but almost essential for $\text{SpO}_2/\text{SpO}_2\text{T}$ alerts:* The log-scale ROC plots in Figure 4 neatly visualize the predictive utility of waveform data for both the RR and $\text{SpO}_2/\text{SpO}_2\text{T}$ alerts. For RR alerts, the plots show some separation between the pipelines with and without access to waveform data, indicating the slight usefulness of waveform data for RR alert classification. In contrast, the plots for oxygen saturation alerts show a much larger gap between plots with and without access to waveform data, with the pipelines having access to waveform data performing much better than those without. The significant predictive utility of waveform data for oxygen saturation alert classification is further substantiated by the ubiquity of waveform features in the top echelon for feature importance, as highlighted in the previous paragraph and Table II

3) *Our WS and FS models perform much better on non-telemetric SpO_2 alerts than telemetric SpO_2 alerts:* All

Alert	GI		PFI	
	WS	Fully Sup.	WS	Fully Sup.
RR	std.resp	q1_rr	std_rr	respHeight
	respHeight	respHeight	q1_resp	std_resp
	mean_rr	mean_rr	respFFT	std_rr
	q1_rr	std_resp	skew_rr	q3_resp
	med_rr	med_rr	respNK1	q1_resp
	plethFFT	kurt_pleth	med_rr	plethTINT
SpO ₂	plethINT	plethFFT	q1_rr	plethTNK1
	plethNK1	plethINT	q3_rr	plethFFT
	kurt_pleth	q3_pleth	plethNK1	plethTFFT
	plethHeight	plethNK1	skew_pleth	med_SpO ₂ T

TABLE II

FEATURE IMPORTANCES CALCULATED FOR RR & SpO₂/SpO₂T ALERTS USING *Gini importance* (GI) AND *permutation feature importance* (PFI) ARE SHOWN ABOVE IN DECREASING ORDER OF IMPORTANCE. THE RANKED FEATURES BETWEEN THE WEAKLY AND FULLY SUPERVISED PIPELINES FOR BOTH ALERT TYPES SHOW SIMILARITIES AND DIFFERENCES IN THE TYPES OF FEATURES USED BY THE RF MODELS FOR EACH SEPARATE PIPELINE.

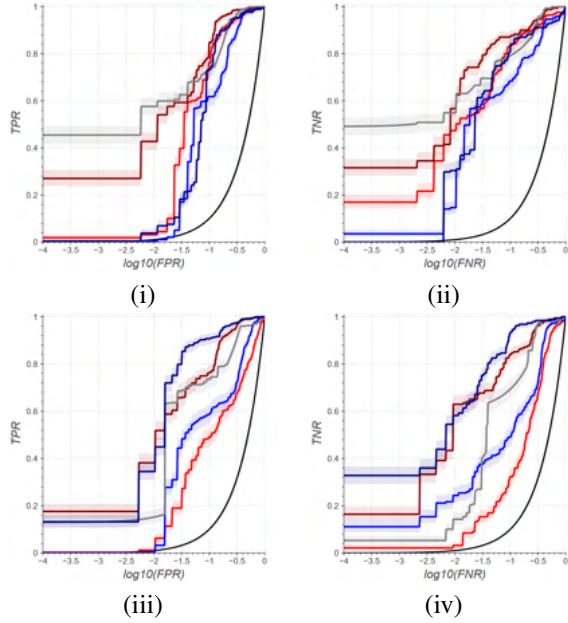


Fig. 4. These 4 log-scale ROC-AUC plots pertain to the ablation experiments conducted on the RR alerts (i & ii) and SpO₂/SpO₂T alerts (iii & iv). Each plot shows the weak supervision pipeline without waveform data (red), with waveform data (dark-red), the fully supervised pipeline without waveform data (blue), with waveform data (dark-blue), and the probability labels pipeline (grey).

pipelines performed well, but both classifiers performed best on the Non-Telemetric data, as indicated by Table III. More importantly, the weak supervision pipeline performed comparably to the fully supervised pipeline in both scenarios. It is unclear if this performance boost is truly due to the splitting of the data or just due to differing class balances, number of data points, or statistical chance, but this a potential avenue for future research.

VI. DISCUSSION

A. Findings

This work has five significant takeaways: (1) The novel core methodologies we developed to derive vital sign numeric values from time series waveform data were reliable, and have meaningful applications beyond the scope of this project. (2) Both the fully supervised and weakly supervised pipelines - when validated on unseen data from a unique patient - remained robust and performed well, with AUC values ranging from 0.898 to 0.964 for all the models. (3) Additionally, the predictive utility of waveform data was found to be only slightly useful for RR alerts, but significantly important for SpO₂/SpO₂T alerts. (4) Based on the real and artifact F1 scores for both alert types, all the classifiers performed better at classifying real alerts correctly, as compared to artifact alerts. This difference in performance is not only important, but also favorable, since in clinical applications, correctly identifying real alerts is a much more desirable result than correctly identifying artifacts. (5) Perhaps most importantly, the weak supervision pipelines were shown to perform at a level on par with their fully supervised counterparts, and for the RR alerts, even outperform them.

B. Limitations and Future Work

There are a few limitations to this work. Firstly, this work assumes a priori knowledge of approximate real versus artifact class balances of vital sign alerts. However, domain experts often already have this knowledge, so these models can still be powerful and reliable tools for classifying vital sign alerts. Secondly, due to the design of our code, the weak supervision pipeline is currently best used as a “fact-checker” that lends a secondary opinion on archived vital sign alert data. In the future, analysis should be carried out to measure the speed and latency of the classification algorithm, before eventually optimizing the design to create and implement a working real-time artifact alert detection system. Despite these limitations, the promising results indicate that a trained WS model could eventually serve as an effective tool for medical practitioners to combat alarm fatigue in the clinic, and it will require vastly less subject matter expert’s effort needed to prepare data to train the models, than previously required.

REFERENCES

- [1] A. Otero, P. Félix, S. Barro, and F. Palacios, “Addressing the flaws of current critical alarms: a fuzzy constraint satisfaction approach,” *Artificial intelligence in medicine*, vol. 47, no. 3, pp. 219–238, 2009.
- [2] L. Chen, A. Dubrawski, D. Wang, M. Fiterau, M. Guillame-Bert, E. Bose, A. M. Kaynar, D. J. Wallace, J. Guttendorf, G. Clermont, *et al.*, “Using supervised machine learning to classify real alerts and artifact in online multi-signal vital sign monitoring data,” *Critical care medicine*, vol. 44, no. 7, p. e456, 2016.
- [3] S. Sendelbach and M. Funk, “Alarm fatigue: a patient safety concern,” *AACN advanced critical care*, vol. 24, no. 4, pp. 378–386, 2013.
- [4] G. Takla, J. H. Petre, D. J. Doyle, M. Horibe, and B. Gopakumaran, “The problem of artifacts in patient monitor data during surgery: a clinical and methodological review,” *Anesthesia & Analgesia*, vol. 103, no. 5, pp. 1196–1204, 2006.

- [5] K. J. Ruskin and D. Hueske-Kraus, "Alarm fatigue: impacts on patient safety," *Current Opinion in Anesthesiology*, vol. 28, no. 6, pp. 685–690, 2015.
- [6] M. Hravnak, T. Pellathy, L. Chen, A. Dubrawski, A. Wertz, G. Clermont, and M. R. Pinsky, "A call to alarms: current state and future directions in the battle against alarm fatigue," *Journal of electrocardiology*, vol. 51, no. 6, pp. S44–S48, 2018.
- [7] K. Saab, J. Dunnmon, C. Ré, D. Rubin, and C. Lee-Messer, "Weak supervision as an efficient approach for automated seizure detection in electroencephalography," *NPJ digital medicine*, vol. 3, no. 1, pp. 1–12, 2020.
- [8] M. Goswami, B. Boecking, and A. Dubrawski, "Weak supervision for affordable modeling of electrocardiogram data," in *AMIA Annual Symposium Proceedings*, vol. 2022. American Medical Informatics Association, 2022.
- [9] R. Sinhal, K. Singh, and M. Raghuwanshi, "An overview of remote photoplethysmography methods for vital sign monitoring," *Computer Vision and Machine Intelligence in Medical Image Analysis*, pp. 21–31, 2020.
- [10] J. Graybeal and M. Petterson, "Adaptive filtering and alternative calculations revolutionizes pulse oximetry sensitivity and specificity during motion and low perfusion," in *The 26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 2. IEEE, 2004, pp. 5363–5366.
- [11] M. He, Y. Nian, and Y. Gong, "Novel signal processing method for vital sign monitoring using fmfcw radar," *Biomedical Signal Processing and Control*, vol. 33, pp. 335–345, 2017.
- [12] J. S. Paul, M. R. Reddy, and V. J. Kumar, "A transform domain svd filter for suppression of muscle noise artefacts in exercise ecg's," *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 5, pp. 654–663, 2000.
- [13] C. Marque, C. Bisch, R. Dantas, S. Elayoubi, V. Brosse, and C. Perot, "Adaptive filtering for ecg rejection from surface emg recordings," *Journal of electromyography and kinesiology*, vol. 15, no. 3, pp. 310–315, 2005.
- [14] G. Lu, J.-S. Brittain, P. Holland, J. Yianni, A. L. Green, J. F. Stein, T. Z. Aziz, and S. Wang, "Removing ecg noise from surface emg signals using adaptive filtering," *Neuroscience letters*, vol. 462, no. 1, pp. 14–19, 2009.
- [15] J. Lee, D. D. McManus, S. Merchant, and K. H. Chon, "Automatic motion and noise artifact detection in holter ecg data using empirical mode decomposition and statistical approaches," *IEEE Transactions on Biomedical Engineering*, vol. 59, no. 6, pp. 1499–1506, 2011.
- [16] R. Couceiro, P. Carvalho, R. P. Paiva, J. Henriques, and J. Muehlsteff, "Detection of motion artifacts in photoplethysmographic signals based on time and period domain analysis," in *2012 Annual international conference of the IEEE engineering in medicine and biology society*. IEEE, 2012, pp. 2603–2606.
- [17] S. K. Bashar, D. Han, A. Soni, D. D. McManus, and K. H. Chon, "Developing a novel noise artifact detection algorithm for smartphone ppg signals: Preliminary results," in *2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*. IEEE, 2018, pp. 79–82.
- [18] D. Pollreisz and N. TaheriNejad, "Detection and removal of motion artifacts in ppg signals," *Mobile Networks and Applications*, pp. 1–11, 2019.
- [19] T. Shimazaki, S. Hara, H. Okuhata, H. Nakamura, and T. Kawabata, "Cancellation of motion artifact induced by exercise for ppg-based heart rate sensing," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2014, pp. 3216–3219.
- [20] M. Hravnak, L. Chen, A. Dubrawski, E. Bose, G. Clermont, and M. R. Pinsky, "Real alerts and artifact classification in archived multi-signal vital sign monitoring data: implications for mining big data," *Journal of clinical monitoring and computing*, vol. 30, no. 6, pp. 875–888, 2016.
- [21] S. Caldas, V. Jeanselme, G. Clermont, M. Pinsky, and A. Dubrawski, "A case for federated learning: Enabling and leveraging inter-hospital collaboration," in *C33. QUALITY, PROCESSES, AND OUTCOMES IN ACUTE AND CRITICAL CARE*. American Thoracic Society, 2020, pp. A4790–A4790.
- [22] J. A. Fries, P. Varma, V. S. Chen, K. Xiao, H. Tejeda, P. Saha, J. Dunnmon, H. Chubb, S. Maskatia, M. Fiterau, et al., "Weakly supervised classification of aortic valve malformations using unlabeled cardiac mri sequences," *Nature communications*, vol. 10, no. 1, pp. 1–10, 2019.
- [23] A. J. Ratner, C. M. De Sa, S. Wu, D. Selsam, and C. Ré, "Data programming: Creating large training sets, quickly," *Advances in neural information processing systems*, vol. 29, pp. 3567–3575, 2016.
- [24] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al., "Scipy 1.0: fundamental algorithms for scientific computing in python," *Nature methods*, vol. 17, no. 3, pp. 261–272, 2020.
- [25] D. Khodadad, S. Nordebo, B. Müller, A. Waldmann, R. Yerworth, T. Becher, I. Frerichs, L. Sophocleous, A. Van Kaam, M. Miedema, et al., "Optimized breath detection algorithm in electrical impedance tomography," *Physiological measurement*, vol. 39, no. 9, p. 094001, 2018.
- [26] D. Makowski, T. Pham, Z. J. Lau, J. C. Brammer, F. Lespinasse, H. Pham, C. Schözel, and S. A. Chen, "Neurokit2: A python toolbox for neurophysiological signal processing," *Behavior Research Methods*, pp. 1–8, 2021.
- [27] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: Rapid training data creation with weak supervision," in *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, vol. 11, no. 3. NIH Public Access, 2017, p. 269.
- [28] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [29] M. Goswami, L. Chen, and A. Dubrawski, "Discriminating cognitive disequilibrium and flow in problem solving: A semi-supervised approach using involuntary dynamic behavioral signals," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, 2020, pp. 420–427.
- [30] M. Goswami, M. Manuja, and M. Leekha, "Towards social & engaging peer learning: Predicting backchanneling and disengagement in children," *arXiv preprint arXiv:2007.11346*, 2020.
- [31] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.
- [32] A. Altmann, L. Tolosi, O. Sander, and T. Lengauer, "Permutation importance: a corrected feature importance measure," *Bioinformatics*, vol. 26, no. 10, pp. 1340–1347, 2010.
- [33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

APPENDIX

Additional tables are included on the following page.

ACKNOWLEDGMENT

I would like to thank Dr. Artur Dubrawski and Mononito Goswami for their extensive guidance and support throughout this summer. In addition, I would like to thank Gus Welter, Dr. Marilyn Hravnak, and Dr. Joo Heung H. Yoon for their time and their helpful advice. I would also like to thank Rachel Burcin, Dr. John Dolan, and Jennie Piotrkowski for their tireless effort to create an enjoyable and fruitful summer research experience for the RISS students. Lastly, I would like to thank the RISS program sponsors, and acknowledge the support of the National Science Foundation (Grant #1659774).

SpO ₂ Alert Type	Pipeline	Real			Artifact			Accuracy	AUC
		Precision	Recall	F ₁	Precision	Recall	F ₁		
Non-Telemetric	Weak Sup.	1.000	0.940	0.969	0.840	1.000	0.913	0.955	0.994
	Fully Sup.	0.985	0.985	0.985	0.952	0.952	0.952	0.977	0.969
Telemetric	Weak Sup.	0.650	0.867	0.743	0.882	0.682	0.770	0.757	0.830
	Fully Sup.	0.800	0.800	0.800	0.864	0.864	0.864	0.838	0.832

TABLE III

PRELIMINARY FINDINGS FROM A NON-CROSS-VALIDATED ANALYSIS ON PURELY TELEMETRIC AND PURELY NON-TELEMETRIC SpO₂ ALERTS SHOW IMPRESSIVE PERFORMANCE FROM THE NON-TELEMETRIC PIPELINES, AND AVERAGE PERFORMANCE FROM THE TELEMETRIC PIPELINES. MORE IMPORTANTLY, THE WEAK SUPERVISION PIPELINE PERFORMS ON PAR, AND FOR SOME METRICS, BETTER THAN, THE SUPERVISED LEARNING PIPELINE FOR BOTH OXYGEN SATURATION ALERT TYPES.

	Respiratory Rate Alerts						Oxygen Saturation Alerts					
	Real			Artifact			Real			Artifact		
	Precision	Recall	F ₁	Precision	Recall	F ₁	Precision	Recall	F ₁	Precision	Recall	F ₁
Weak Sup.	0.917	0.973	0.944	0.909	0.754	0.824	0.896	0.938	0.916	0.840	0.751	0.793
Majority Vote	0.959	0.838	0.895	0.667	0.901	0.766	0.911	0.947	0.929	0.866	0.788	0.826
Fully Sup.	0.897	0.954	0.925	0.844	0.696	0.763	0.952	0.907	0.929	0.809	0.894	0.849
Prob. Labels	0.911	0.948	0.929	0.836	0.743	0.786	0.838	0.961	0.895	0.865	0.577	0.692
WS w/o WF	0.902	0.950	0.925	0.836	0.714	0.770	0.801	0.773	0.787	0.520	0.561	0.540
Sup. w/o WF	0.891	0.889	0.890	0.692	0.696	0.694	0.837	0.759	0.796	0.546	0.661	0.598
Maj. w/o WF	0.943	0.763	0.844	0.569	0.871	0.688	0.807	0.752	0.778	0.509	0.587	0.546

TABLE IV

WE CALCULATED VARIOUS PERFORMANCE METRICS OF ML PIPELINES ON THE CLASSIFICATION OF RR & SpO₂/SpO₂T ALERTS AND FOUND THAT THEY ALL PERFORMED REALLY WELL. MOST INTERESTING, WAS THE WEAK SUPERVISION PIPELINE'S COMPARABLE, AND IN SOME CASES, SUPERIOR, PERFORMANCE OVER THE FULLY SUPERVISED LEARNING PIPELINE FOR BOTH ALERT TYPES.

	Respiratory Rate Alerts						Oxygen Saturation Alerts													
	Accuracy	AUC	FPR	50% TPR	FNR	50% TNR	TPR	1% FPR	TNR	1% FNR	Accuracy	AUC	FPR	50% TPR	FNR	50% TNR	TPR	1% FPR	TNR	1% FNR
Weak Sup.	0.915	0.951	0.012	0.008	0.428	0.567	0.881	0.940	0.011	0.009	0.382	0.630								
Majority Vote	0.855	0.952	0	0.015	0.551	0.409	0.899	0.951	0.011	0.007	0.458	0.630								
Fully Sup.	0.886	0.898	0.07	0.023	0.038	0.304	0.903	0.964	0.016	0.007	0.345	0.582								
Prob. Labels	0.894	0.936	0.006	0.002	0.577	0.550	0.844	0.902	0.016	0.037	0.151	0.143								
WS w/o WF	0.887	0.918	0.035	0.010	0.031	0.474	0.709	0.754	0.111	0.197	0.012	0.032								
Sup. w/o WF	0.838	0.871	0.053	0.019	0.004	0.146	0.730	0.825	0.037	0.106	0.002	0.243								
Maj. w/o WF	0.792	0.899	0.07	0.019	0.080	0.199	0.702	0.779	0.058	0.167	0.025	0.069								

TABLE V

THE PERFORMANCE METRICS OF *practical utility* FOR THE ML PIPELINES CLASSIFYING RR & SpO₂/SpO₂T ALERTS WERE ALSO CALCULATED. WE AGAIN FOUND THE WEAK SUPERVISION PIPELINE OUTPERFORMED, OR AT THE VERY LEAST, MATCHED THE PERFORMANCE OF THE FULLY SUPERVISED LEARNING PIPELINE FOR BOTH ALERT TYPES.

Tree Modeling for Robotic Manipulation using a 3D Autoencoder

Christian Eberle¹ and Oliver Kroemer²

Abstract—Modeling tree structures is a crucial requirement for performing manipulation tasks on trees, such as pruning or harvesting. One big challenge in modeling these structures is the visual occlusion through twigs and leaves. Therefore, the goal of this work is to reconstruct the 3D model of a tree in full leaf from partial view, using a 3D autoencoder neural network. Our contribution is to provide a method of generating training data for such an autoencoder, based on 3D models of trees grown in simulation. We provide both ground truth models as well as a models that were reconstructed from the images of a kinect camera. The images were obtained in the robot simulation software CoppeliaSim.

Index Terms—Agricultural Automation, Perception for Grasping and Manipulation, RGB-D Perception, Simulation and Animation

I. INTRODUCTION

At present, almost all manipulation tasks involved in maintaining and harvesting crops are done by hand. Not only is this physically demanding labor, it is often unhealthy work in the long term. As qualified human workers in modern agriculture also become rarer and more expensive, there is an increasing need for automation in this field. Thus, a lot of current research effort explores the development of agricultural robots that can physically interact with its environment [1], [2].

For tree-based crops such as apples, the important manipulation tasks to automate are pruning and harvesting. Furthermore, monitoring fruitlets for diseases or to predict next season's fruit load often requires pushing aside branches. Most commercial robots designed for these tasks still require human supervision to operate. Some fruit picking robots do not need human supervision but are limited in the environment in which they can operate, thus requiring extensive effort from the farmer to prune the fruit trees in a way that the robot can operate on them [3]. To perform these tasks, a robot must incorporate a perception component that models the 3D branch structure of the tree. This component is crucial to reach a goal state in the tree when performing manipulation tasks. However, modeling branch structures comes with many challenges: Trees are highly complex objects with a lot of variation in shape, size and surface properties such as color and texture. Additionally, one always has to adapt to different lighting conditions. But most importantly, the branch structure is usually not directly visible because leaves and smaller twigs obstruct the view. In a comprehensive

review on Computer Vision for fruit harvesting robots [4], Kapach et al. identified the occlusion from foliage as “the main challenge that our research community faces today”.

The goal of this work is to tackle the occlusion problem by training a 3D autoencoder neural network to do infilling on the occluded regions of the scene. To do this, it would first have to segment the voxel grid into regions of unoccupied voxels, surface voxels and occluded voxels. Our contribution is to provide a pipeline for generating training data. This data is based on simulation and provides both ground truth 3D data as well as 3D data from the partial view of a kinect camera.

The following sections are organized as follows: Section II will explore related works, followed by a section on the method of generating data. In the Results section, we will present the dataset we created before ending with a section discussing potential future works.

II. RELATED WORK

Modeling trees is a task that many researchers work on for various reasons and the approaches for solving this task can be roughly divided into two categories: The first category includes approaches utilizing traditional, procedural algorithms. The other category of approaches utilize machine learning algorithms to find a solution based on given data.

A. Procedural Approaches

In computer graphics, tree models are often generated by simulating the growth over time. Algorithms simulating tree growth are commonly based on Lindenmayer-systems (L-systems), a formal grammar used to describe organisms with complex branching structures [5]. A Lindenmayer system is a set of rules that is iteratively applied to a set of starting symbols, thus making use of the recursive nature of cell-based organisms. While methods like L-systems work well for creating realistic-looking tree models in animation, agricultural applications instead need to model existing trees accurately using sensory information. In a recent paper [6], Yandun et al. proposed two approaches for this task: The first approach relies on the Space Colonization Algorithm (SCA) to reconstruct the 3D-model from the images of a stereo camera. Like L-systems, SCA is a recursive algorithm and it builds the tree model based on a number of attraction points at which the tree has potential to further grow [7]. To obtain the attraction points, Yandun et al. used a deep neural network for object recognition called Faster-RCNN. The second approach involves Laplacian-based contraction (LBC), an algorithm that can extract the skeleton from discrete geometric data such as point clouds. For synthetic

¹Christian Eberle is with the Department of Computer Science, University of Tübingen, Germany.
christian-thomas.eberle@student.uni-tuebingen.de

²Oliver Kroemer is with The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. okroemer@cmu.edu

as well as real trees, LBC could reconstruct the 3D shape more accurately than SCA and was shown to be a promising approach.

B. Learning Approaches

Machine learning methods have shown to be a powerful tool for agricultural machine vision systems [8]. Therefore, many other approaches use machine learning methods to detect or extract features from tree imagery: Multi-class support vector machines (SVM) were used to detect fruits and branches for a fruit harvesting robot designed for citrus trees [9]. SVMs project data into higher-dimensional space to separate it into multiple classes with a hyperplane. In this case, the SVM classified the pixels of an RGB image as either citrus, branch, leaf or background. Other research instead focuses on deep learning approaches, particularly convolutional neural network (CNN) architectures. CNN's are a popular architecture and have shown to be highly effective in many image processing tasks. Notably, Zhang et al. used color and depth data from a Kinect camera to detect branches in apple orchards with a Regions-Convolutional Neural Network (R-CNN) [10].

Another architecture that has also proven to be useful is the autoencoder neural network. This architecture consists of two components: First, an encoder component transforms the input into a compressed representation. This component can be described by a function $e(x)$ mapping its input to a lower-dimensional space. In this function, the input x is first multiplied by a set of weights, described by the weight matrix W_e . Next, the biases b_e of the encoder network are added before passing it through an activation function ϕ :

$$e(x) = \phi(W_e^T x + b_e) \quad (1)$$

A decoder network will then transform this compressed representation $h(x)$ back into its original dimensionality to form the output. The decoder network has its own weight matrix W_d , biases b_d and activation function ϕ , but the resulting function $d(e(x))$ looks similar to $h(x)$:

$$d(e(x)) = \phi(W_d^T e(x) + b_d) \quad (2)$$

In a recent paper, a deep convolutional autoencoder was used for feature extraction on images of plant leaves [11]. These features supported disease identification of apple, cherry and corn plants and led to up to 98.8% classification accuracy. Furthermore, autoencoder architectures were also shown to be effective not only on images but also on three dimensional data: To tackle the problem of 3D shape retrieval, autoencoder architectures have been successfully used to reconstruct the 3D shape from a depth image [12] or 2D images [13]. [12] achieved near state-of-the-art performance in a collection of benchmark datasets, such as the Princeton Shape Benchmark (PSB).

III. METHODS

Our data generation pipeline was structured as follows: First, we created an assortment of artificial trees the 3D animation tool grove3D. Next, we exported those trees into the

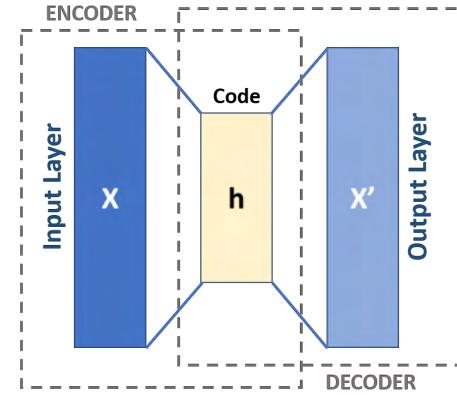


Fig. 1: Overview of an autoencoder architecture. The input X is transformed by the encoder into h , a compressed representation. Next, the decoder transforms h back into the original dimensionality and outputs X' .

robot simulation software CoppeliaSim, where sensory data of the trees was generated. Finally, we converted the sensory data into a voxelized representation of the tree. The ground truth is a voxelized representation based on the original mesh file of the tree model. This pipeline provides both training and teacher data, thus making supervised learning possible.

A. Generating artificial trees

All 3D models were generated in Blender, a 3D computer graphics software [14]. To generate realistic tree models, we utilized the latest release (release 10) of The Grove 3D, a Blender add-on [15]. The Grove 3D was designed specifically for this purpose and includes tools to "grow" a variety of tree species. It does so by simulating the relevant genetics and mechanism in natural tree growth, such as Phyllotaxis or Photosynthesis. The resulting 3D models were exported from Blender as wavefront object files. This data structure represents an object as a mesh containing a collection of edges, vertices and faces.

B. Sensor Simulation

Next, each 3D tree model was imported into a scene of CoppeliaSim, a program for robot simulation [16]. The wavefront object files did not include texture, thus the tree model was not colored. Not including color in the scene might make the task of reconstruction more difficult for the autoencoder, since this information might be useful to distinguish between green leaves and brown branches. However, not relying on color also has its advantages: In real scenes, color is an instable feature that may be misleading due to varying lighting conditions [17]. Therefore, not relying on color features for segmentation and infilling might make the autoencoder more robust in real-life applications.

As sensor, we used a kinect camera that provides both an rgb and a depth image. Kinect cameras are low-cost sensors, making them attractive for commercial use. The depth camera of a kinect sensor relies on the time-of-flight (TOF) mechanism. TOF sensors calculate depth by



Fig. 2: Sample 3D model of a walnut tree, in Blender and a CoppeliaSim scene.

On the right hand side, b) also shows the position of the kinect camera as well as the rgb image (top left) and depth image (top right)

transmitting light and then measuring the time the light takes to travel to a given location [18]. Further image processing was done with Open3D, a library for dealing with 3D data [19].

From the rgb and depth images of the kinect, we created a voxelized representation of the tree. To do this, we first created a point cloud based on the rgb and depth image and subsequently converted the point cloud to a voxel grid. This voxel grid is a good approximation of sensory data that might occur in a similar scene in a natural environment. As seen in Fig. 2, the kinect takes the images from a single position and the resulting images present only a partial view of the scene. Therefore, it contains the same occlusion in view as in a real scene, or at least a very similar one. The voxelized representation of the tree obtained from the kinect images provides the surface area visible from that view angle. To go from this representation to a complete 3D model of the tree, one has to do infilling on the occluded voxels behind the surface voxels.

Another thing that should be considered is that the conversion process from image to point cloud to voxel grid will lead to some loss in information. As the voxel size is chosen to be relatively large, the resolution of the 3D model is decreased. Increasing the resolution would make the task easier but as a tradeoff leads to a much higher computational cost in data collection and training. We chose a voxel size of 10cm, since our computational resources were limited and we wanted to generate large quantities of data.

C. Ground Truth

To obtain the ground truth tree models, we loaded the wavefront object files into Open3D and converted them into a voxel grid. To be able to directly compare the voxel grid obtained from sensory data to the ground truth voxel grid, the voxel size is kept the same.

IV. RESULTS

Our dataset consisted of young trees that were 3 years old. Young trees have the advantage that their shape is less

complex than older trees, thus providing a good starting point for first experiments in training a 3D autoencoder neural network. The young age was also a pragmatic decision, since the resulting files were smaller and subsequent processing came with a lower computational cost. We chose walnut trees because we are ultimately aiming for agricultural applications and walnuts are high-value crops. Additionally, they were the only tree species available in grove3D that included fruits. The walnut trees were grown using the default parameters and the corresponding twig package "Walnut". Default parameters allow the trees to grow without constraints, as they would in the wild. To simulate the environment of a maintained orchard in more detail, one could explore the tools provided by grove3D for auto-pruning. However, since we're only using very young trees auto-pruning doesn't make a big difference yet.

In total, we generated 500 mesh models of young walnut trees. From each mesh model, we created both a ground truth voxel grid as well as a voxel grid that was reconstructed from rgb and depth image.

V. DISCUSSION

This work provides a pipeline for generating simulated tree data that can be used to train a 3D autoencoder neural network to reconstruct the 3D model of a tree from the images of a kinect camera. Solving this task would significantly contribute to the development of agricultural applications related to robotic manipulation on trees. The next step is to implement the 3D autoencoder neural network and to train it using the data we provided. Furthermore, future work could expand on our data generation pipeline. Both of these possibilities will be explored in the following two sections.

A. Autoencoder architectures

The task of this 3D autoencoder neural network would be divided into two subtasks. The first task is a segmentation problem, where the voxel grid of the tree would be segmented into unoccupied regions, surface regions and

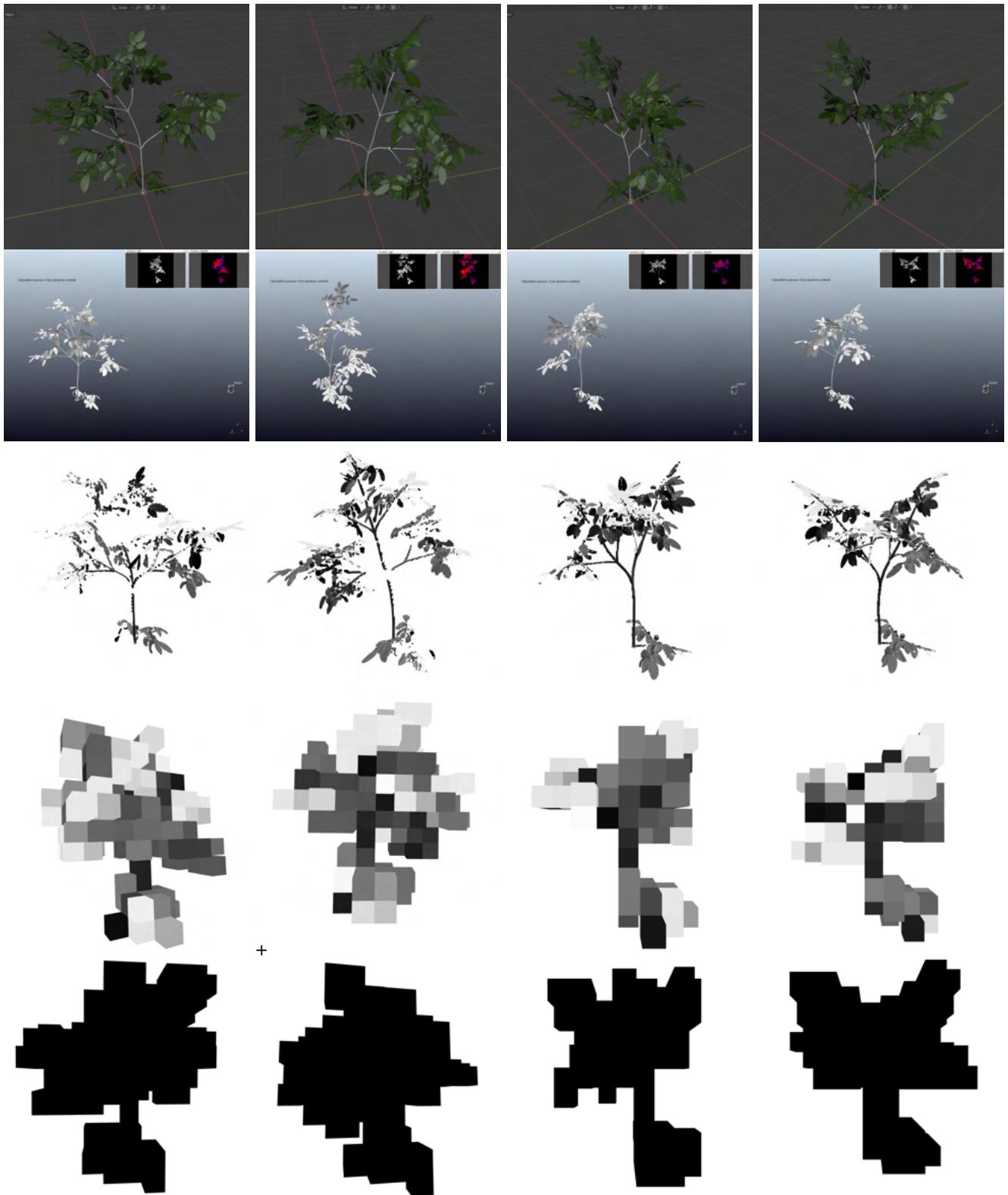


Fig. 3: Four sample trees from the dataset.

From top to bottom: Tree in blender, scene in CoppeliaSim, point cloud, voxel grid, ground truth voxel grid

occluded regions. The second task would consist of filling in voxels within the regions identified as occluded in the first step.

A question that remains open is which exact architecture to choose, since there is a variety of different autoencoder architectures. [20] uses contractive autoencoders for object-

specific 3D reconstruction. Contractive autoencoders have the advantage of providing invariance during feature extraction which makes the network more robust to noise. While the main goal of [20] is shape retrieval and the 3D reconstruction is only an intermediate step, the architecture is well-suited for our task. Moreover, [20] also uses depth images from a kinect camera as input.

Variational autoencoders (VAEs) have also shown great potential in similar tasks. VAEs apply regularisation during training, resulting in an encoding that has especially suitable properties for generating new data. Since we are generating new data when filling in occluded voxels, the VAE architecture is a strong candidate for future work.

B. Expanding the data generation pipeline

To expand on the data generation pipeline one could grow older trees, resulting in more complex models. While this requires more computational resources and makes the task of 3D reconstruction more challenging, models of older trees would be a more faithful reproduction of trees in a commercial setting. Additionally, almost all commercial orchards also prune their trees to maximize yield. This could also be simulated using the auto-pruning tools provided by grove3D. Another way to expand on our data generation pipeline would be to add textures in the CoppeliaSim scene, thus providing the autoencoder with color features. As discussed earlier, including color might make the task easier in simulation but runs the risk of leading to less robustness in real-world applications due to variance in lighting conditions. Furthermore, the resolution of the voxelized representations could be improved by decreasing the voxel size. A higher resolution will depict the tree shape more accurately but comes with a higher computational cost because a larger set of voxels needs to be processed. Ultimately, the level of accuracy required from the model will depend on the application. For example, pruning often involves precise cuts and therefore requires a very accurate 3D model of the tree.

ACKNOWLEDGMENT

Special thanks to Rachel Burcin and John Dolan for coordinating the RISS program and to Oliver Kroemer for his great support and mentoring. Christian Eberle would also like to thank the DAAD for funding this work.

REFERENCES

- [1] L. He and J. Schupp, "Sensing and automation in pruning of apple trees: A review," *Agronomy*, vol. 8, no. 10, 2018. [Online]. Available: <https://www.mdpi.com/2073-4395/8/10/211>
- [2] Q. Zhang, M. Karkee, and A. Tabb, "The use of agricultural robots in orchard management," *Robotics and automation for improving agriculture*, J. Billingsley, Ed. Burleigh Dodds Science Publishing, 2019, pp. 187–214. [Online]. Available: <http://dx.doi.org/10.19103/AS.2019.0056.14>
- [3] J. Zhang, L. He, M. Karkee, Q. Zhang, X. Zhang, and Z. Gao, "Branch detection for apple trees trained in fruiting wall architecture using depth features and regions-convolutional neural network (r-cnn)," *Computers and Electronics in Agriculture*, vol. 155, pp. 386–393, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0168169918304162>
- [4] K. Kapach, E. Barnea, R. Mairon, Y. Edan, and O. Ben-Shahar, "Computer vision for fruit harvesting robots – state of the art and challenges ahead," *Int. J. Computational Vision and Robotics*, vol. 3, pp. 4–34, 2012.
- [5] A. Lindenmayer, "Mathematical models for cellular interactions in development ii. simple and branching filaments with two-sided inputs," *Journal of Theoretical Biology*, vol. 18, no. 3, pp. 300–315, 1968. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/0022519368900805>
- [6] F. Yandun, A. Silwal, and G. Kantor, "Visual 3d reconstruction and dynamic simulation of fruit trees for robotic manipulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.
- [7] A. Runions, B. Lane, and P. Prusinkiewicz, "Modeling trees with a space colonization algorithm," *NPH*, vol. 7, pp. 63–70, 2007.
- [8] T. U. Rehman, M. S. Mahmud, Y. K. Chang, J. Jin, and J. Shin, "Current and future applications of statistical machine learning algorithms for agricultural machine vision systems," *Computers and Electronics in Agriculture*, vol. 156, pp. 585–605, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0168169918304289>
- [9] L. Qiang, C. Jianrong, L. Bin, D. Lie, and Z. Yajing, "Identification of fruit and branch in natural scenes for citrus harvesting robot using machine vision and support vector machine," *International Journal of Agricultural and Biological Engineering*, vol. 7, no. 2, pp. 115–121, 2014.
- [10] J. Zhang, L. He, M. Karkee, Q. Zhang, X. Zhang, and Z. Gao, "Branch detection for apple trees trained in fruiting wall architecture using depth features and regions-convolutional neural network (r-cnn)," *Computers and Electronics in Agriculture*, vol. 155, pp. 386–393, 2018.
- [11] K. Trang, L. TonThat, and N. G. Minh Thao, "Plant leaf disease identification by deep convolutional autoencoder as a feature extraction approach," in *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, 2020, pp. 522–526.
- [12] Z. Zhu, X. Wang, S. Bai, C. Yao, and X. Bai, "Deep learning representation using autoencoder for 3d shape retrieval," *Neurocomputing*, vol. 204, pp. 41–50, 2016, big Learning in Social Media Analytics. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925231216301047>
- [13] J. Zhang, K. Li, Y. Liang, and N. Li, "Learning 3d faces from 2d images via stacked contractive autoencoder," *Neurocomputing*, vol. 257, pp. 67–78, 2017, machine Learning and Signal Processing for Big Multimedia Analysis. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925231217301431>
- [14] <https://www.blender.org/about/>, Aug. 2021.
- [15] <https://www.thegrove3d.com/>, Aug. 2021.
- [16] <https://www.coppeliarobotics.com/>, Aug. 2021.
- [17] G. D. Finlayson, "Colour and illumination in computer vision," *Interface focus*, vol. 8, no. 4, p. 20180008, 2018.
- [18] M. Vázquez-Arellano, H. W. Griepentrog, D. Reiser, and D. S. Parafos, "3-d imaging systems for agricultural applications—a review," *Sensors*, vol. 16, no. 5, p. 618, 2016.
- [19] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.
- [20] J. Feng, Y. Wang, and S.-F. Chang, "3d shape retrieval using a single depth image from low-cost sensors," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016, pp. 1–9.
- [21] F. Y. Narvaez, G. Reina, M. Torres-Torriti, G. Kantor, and F. A. Cheein, "A survey of ranging and imaging techniques for precision agriculture phenotyping," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 6, pp. 2428–2439, 2017.

SR-UVOS: Sparse Reconstruction for Unsupervised Video Object Segmentation

Daniel Ekpo¹, Senthil Purushwalkam², Abhinav Gupta³

Abstract—With applications in autonomous driving, human-robot interaction, video editing, etc. this is a very compelling task with a lot of attention in the last few years. Although a lot of current methods claim to be unsupervised, they still use some supervision from other datasets. This kind of supervision is rather expensive, does not scale well, and the methods are not easily transferable to other domains. We present a novel algorithm to tackle this task without using any human annotation. We build from recent progress in vision transformers and knowledge distillation to build a model that is aware of time and space correspondences between video frames. We hypothesize that learning both intra-frame and inter-frame correspondences with the transformer self-attention modules can improve the accuracy on video object segmentation. Vision transformer self-attention modules have been shown by recent work to be able to pay attention on image regions with foreground objects. To this end, we leverage recent work on video understanding with transformers. Specifically, we use the DINO model proposed by [1] as a base model and build on it to learn spatial and temporal correspondences between video frames. We use knowledge distillation to train a student model that must learn to match the teacher network’s output. To make this task non-trivial and encourage the network to learn semantic correspondences between frame pairs, we add a reconstruction module which learns to reconstruct the embeddings for the overlapping region between the images in the frame pair. We show in our experiments that our simple self-supervised algorithm is enough to learn semantic visual representations in video frames and semantic intra-frame correspondences between video frames.

Index Terms—self-supervised learning, video object segmentation

I. INTRODUCTION

With applications in autonomous vehicles, human-computer interaction, video editing, the task of video object segmentation has received an increased attention in recent years. The task involves localizing foreground objects in a video and putting segmentation masks over those objects. A challenging sub-task is separating foreground objects from the background - a rather ill-posed challenge since a “foreground” object is not well defined. A common definition among researchers is that foreground objects are salient objects that would normally get a human viewer’s attention [2] when watching the whole video sequence [3]. The background is defined as uninteresting “stuff” like sky, buildings, etc. The task is difficult since the model does not have any prior knowledge about the foreground objects. Video

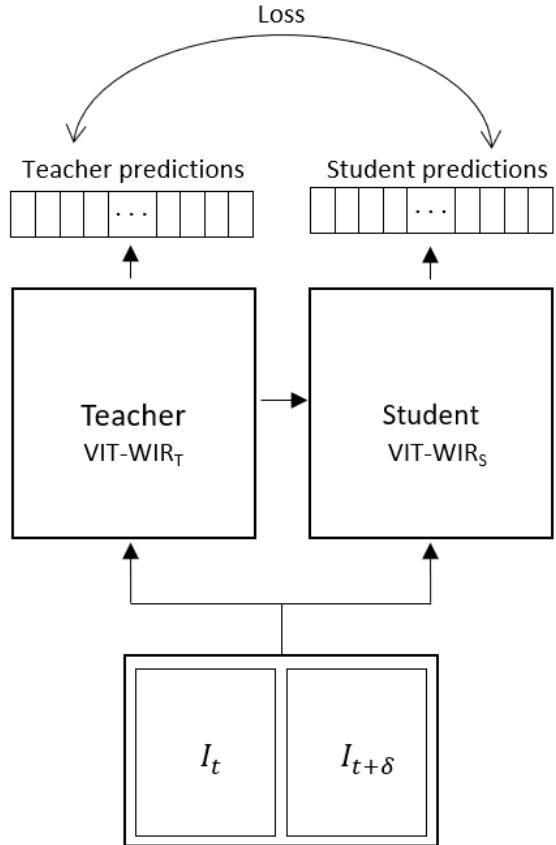


Fig. 1. SR-UVOS: Each network - the teacher and student take a pair of images and an optional reference embedding as input. We compare the final predictions to calculate the prediction loss

object segmentation (VOS) can be divided into two tasks - (weakly) supervised video object segmentation and unsupervised video object segmentation (UVOS). In the supervised case, the model receives the segmentation masks for the first frame which is then used as a prior for subsequent frames. In unsupervised video object segmentation, as defined by [3], the segmentation mask for the first frame is not given, so the model must learn to accurately segment foreground objects in the first and subsequent frames.

It is also important to point out that some methods [4], [5], [6] focus on segmenting a single foreground object. Some other methods [7] attempt to segment all the foreground objects in each video frame. The Davis challenge [3] does not penalize models for predicting too many non-

¹ Daniel Ekpo is a student at Brigham Young University

² Senthil Purushwalkam is a PhD student at the Carnegie Mellon Robotics Institute

³ Dr. Abhinav Gupta is an associate professor at the Carnegie Mellon Robotics Institute

overlapping object segmentation masks. In this work, we tackle unsupervised video *multi-object* segmentation. This is a more challenging problem since the model must discriminate between foreground objects and the background as well as discriminate between multiple class-agnostic objects.

The lack of any priors on the segmentation mask makes UVOS more challenging than its supervised counterpart. Since a common strategy in VOS is to propagate the mask predictions from previous frames when making predictions on the current frame [1], [6], inaccurate predictions in the first frame can quickly be propagated to other frames, degrading the performance of the model. To tackle this problem [8] use the so-called Selector Net module to propagate only high quality masks out of a several possible mask predictions. Background and foreground similarity (similarities in color and texture) is another problem that plagues UVOS. This problem occurs when the foreground object and the background are very similar in color or texture, making it difficult to use appearance properties only to localize foreground objects. To overcome this problem, some methods [9], [10], [11] use optical flow predictions to augment the appearance features. Computing the optical flow predictions is rather expensive, therefore increasing the amount of computation and time needed for training and evaluation. In addition, some foreground object stay stable or don't move enough between frames for optical flow predictions to be useful for detecting foreground objects.

One could be tempted to treated UVOS as a special case of object segmentation and simply treat each frame independently, effectively reducing the problem to the task of object segmentation in still images. While this has been shown to work to some extent [12], [13], [14], models that follow this approach miss the rich temporal information between video frames. Both spatial and temporal correspondences between frames can provide rich information about the current objects, providing useful priors for the current frame. Some methods compute the optical flow between consecutive frames and use that for the temporal correspondence, but as we pointed out earlier, this can be slow and inaccurate when the foreground is static [11]. The vision transformer [15] which has been successfully used in different vision tasks, provides an efficient way to compute the spatio-temporal relationship between two video frames.

We start by asking if we can learn a model to perform better than the current state-of-the-art methods without using motion information via optical flow, or any expensive post-processing step. This question was inspired by the intuition that frames that are separated by a close enough frame distance δ should share similar objects - that is there should be a level of consistency between the two frames. This observation leads to asking if computing the correlations between the two frames would provide enough priors to improve the segmentation accuracy.

Inspired by the recent work on visual representation learning with vision transformers by [1], we learn a model to compute correspondences between video frames using the vision transformer's [15] attention module. This alleviates

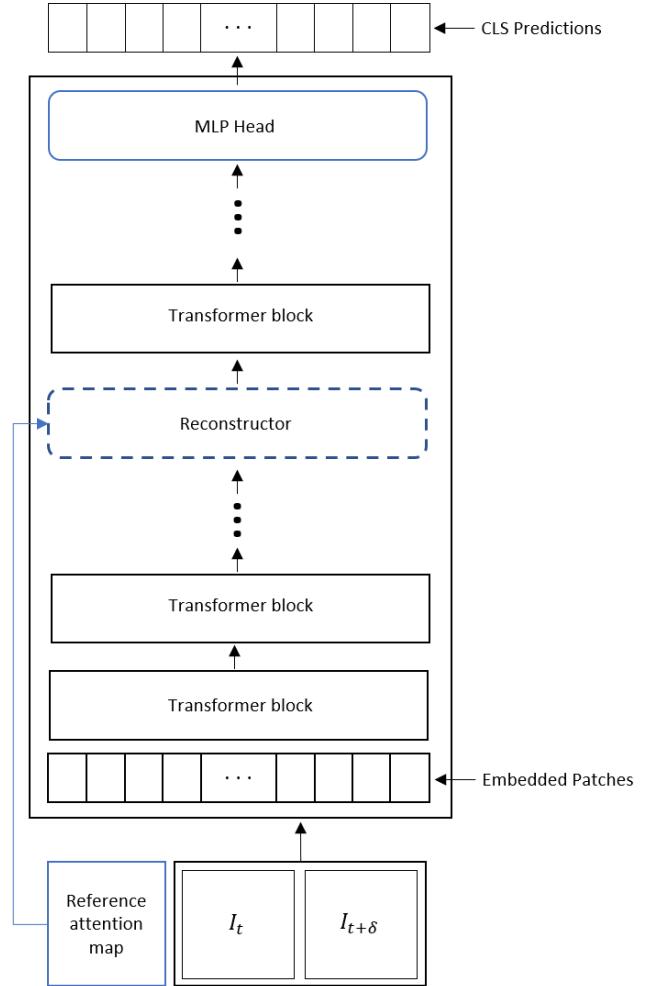


Fig. 2. VIT-WIR: Vision transformer with reconstruction takes a pair of images as input and an optional reference embedding for reconstruction. The reconstruction is done in the reconstruction module and its output is passed to the next transformer block

the need to compute the optical flow or use any complex method. Instead of creating a pretext task, as is popular in recent unsupervised methods, we use knowledge distillation, similar to [1]. to train a student network to learn to match the teacher networks predictions. We evaluate our model on the DAVIS 2019 dataset.

Our main contributions are as follows:

- We propose and end-to-end, simple, and efficient fully unsupervised method to tackle the video *multi-object* segmentation task. We discuss our method in Section 3
- We introduce a modification to the vision transformer which includes adding a reconstruction module to reconstruct the embeddings for the patches in the overlapping region between the images in the frame pair

II. RELATED WORK

A. Self-supervised learning

A large body of work in this domain focus on learning useful task-independent discriminative features from images and

videos. The weights for these models are then used directly [1] or fine-tuned [16], [17] on a specific task to solve the given problem. Often these models perform decently well, but are usually subpar to supervised methods that are trained for the specific task. In our work, we focus on learning a model with no human annotation to perform significantly well on the specific task of video object segmentation.

Recent approaches use a pretext task that force the model to learn discriminative features to be able to do well on the given task. These methods typically use contrastive learning where the training data is grouped into positive and negative pairs. The task then is to learn a feature space where positive pairs are close together and negative pairs are far apart by at least some delta. Chopra et al [18] propose a similarity metric function that maps input images to points in a low dimensional target space such that the L_1 distance between objects of the same category is minimized and the distance between objects in different categories is maximized. Doersch et. al [19] introduce a context prediction pretext task for visual representation learning. In their formulation, the model learns to predict the position of an image patch in relation to a query image patch. They show that the model must learn to recognize objects and their parts to do well on this task. Others [17], [20], [21] follow this practice by introducing different pretext tasks for unsupervised visual representation learning. Although our work focus on visual representation learning, our approach does not follow this paradigm. Instead, we use knowledge distillation to train a target student network to match the features of teacher network.

Most related to our work is the DINO model proposed by Doersch et al. [1]. They build on the vision transformer [15] and knowledge distillation [22] by training a student transformer to learn to match the predictions of the teacher vision transformer model. To encourage the model to learn useful visual representations, the teacher network is shown two global crops from the image and the student sees the global crops as well as multiple local crops. In contrast to other common knowledge distillation methods, where the student is trained after the teacher, the authors train the teacher simultaneously with the student. The student transformer's weights are updated using stochastic gradient descent, while the teacher transformer's weights are update from the student network's weights using momentum update [23]. Our work differs from their work in that we do not have local crops, and instead of taking two global crops from the same image, we use two images from the same video with a frame distance δ as our global crops. Another important difference between our work and theirs is that we add a reconstruction module in the transformer to discourage the model from finding a trivial solution while encouraging it to focus on the overlapping areas in the frame pairs.

B. Unsupervised Video Object Segmentation

The task of unsupervised video object segmentation (UVOS) is a rather new task introduced in the DAVIS 2019 challenge [3] and has received some attention in the recent

years. Some approaches [8], [12], [14] rely on bounding bounding box predictions from off-the-shelf object detection algorithms like Mask R-CNN [24]. These models, as expected, are not end-to-end trainable and do not take advantage of the temporal correlation between video frames since each frame is treated as an independent entity.

Liu et al [25] try to tackle the background and foreground similarity problem in UVOS by introducing the F2Net model which first predicts a center point for the object of interest. The model leverages this predicted center point of the foreground object as a spatial guidance prior to encourage the segmentation module to focus on the primary object. Inspired by how humans tend to segment objects, the authors hypothesize that this prior can help the segmentation module focus on the pixels of the foreground object, resulting in more accurate segmentation masks. In addition, they propose an attention-based so-called dynamic information fusion module to dynamically select the most discriminative of three features (inter-frame, intra-frame and original semantic features) to use for the final object segmentation masks. Their approach works decently well for single objects, however they did not show any results with multiple objects, and did not mention how their approach could be used for multi-object scenarios. Also, their architecture is rather complex.

Garg et al [8] attempt to tackle the problem of mask error propagation/drifting between frames by using a quality discrimination module to select the best masks for each frame from an ensemble of segmentation mask predictions. They show from experiments that this module improves the object segmentation mask predictions. However, they rely on region of interest predictions from Mask R-CNN [24] so their model is not end-to-end trainable.

In this category, our work is most related to [26] and [27] who use vision transformers to tackle the UVOS task. However, our method is different from theirs since their methods are supervised. Our approach focuses on learning rich semantic visual representations and correspondences first, and then adapting the model for the UVOS task.

C. Self-attention

Early approaches, mostly in natural language processing [28], [29] used recurrent neural networks to compute correspondences and help the model know what parts of the sentence to focus on (attention). Recurrent neural networks have been adopted in vision tasks [30], [31], [32], [33]. Wang et al [34] proposed a general self-attention algorithm for capturing the affinity between image patches or whole images. Recent approaches towards attention have adopted the transformer model [28]. [15] introduced the vision transformer, which uses the same architecture as [28] but uses image patches instead of word embeddings as the transformer input. Vision transformers have been applied with success to different computer vision tasks including video classification [35], video instance segmentation [36], object tracking [37], and video object segmentation [27] [26]. Our work builds on this recent successes as we use the vision transformer's self-attention mechanism to compute correspondences between

video frames efficiently.

D. Knowledge distillation

Knowledge distillation, is the process of training a - usually smaller network to match the output of a - usually - bigger network. The student network learns to match the teacher network's logits. Introduced by Hinton et al. [22], this approach has successfully been used for model training and compression [38], [39], [40]. In our work, we use knowledge distillation to train a student vision transformer model to match the outputs of a teacher model of the same architecture.

III. METHOD

We do not directly train our model on the task of video object segmentation. Going with popular methods, we first train our model to learn useful semantic visual representations and inter-frame correspondences and then test on the DAVIS [3] dataset. Our model learns to solve two problems - one, to accurately reconstruct the embeddings for the overlapping area of the images in the frame pair, two, to match the teacher network's predictions.

We define $I_t, I_{t+\delta}$ as two frame images from the same video $V = \{I_0, I_1, \dots, I_N\}$ where N is the total number of frames in the video. I_t is randomly sampled from V at time step t and $I_{t+\delta}$ is δ time steps away from I_t . We define a single training example as $x = \{I_t, I_{t+\delta}\}$ such that $x \subset V$. Let g_{θ_s} and g_{θ_t} be the student and teacher transformer networks respectively, parameterized by θ_s and θ_t respectively.

A. The reconstructor

Both networks have the same architecture with a set of transformer blocks. Let the student network's transformer block be $B_s = \{b_{s0}, b_{s1}, \dots, b_s M\}$ where M is the total number of transformer blocks. Each transformer block encodes an embedding from its input, we define the embedding for the training example at block i as $b_{si}(x)$. The reconstructor takes the embedding at the same transformer blocks from the teacher and student models and try to reconstruct the student embedding for the part of the overlapping part of both images. Let o be the coordinates where both image crops overlap. This is computed in the dataset. We mask a certain percent p of the overlapping region between the two image crops and let the reconstructor reconstruct the masked attention maps. We define the reconstructor part of the attention:

$$z = R(b_{si}, b_{ti}, o, p)$$

We set i as a hyperparameter. In practice R is a simple transformer model with only one transformer block. We replace the masked part of b_{si} with z and pass it to the next transformer block. In our experiments we try different values for p including 0.25, 0.50, 0.75. We compare the reconstruction with the ground truth and compute the reconstruction loss using the mean squared loss given as:

$$L_{recon} = \frac{1}{n} \sum_{i=1}^n (e_i - z_i)^2$$

B. Full model

Our method, SRU-VOS, shares the same general architecture as DINO [1] with the vision transformer modified to include a reconstruction module. The overall architecture is shown in figure 1. 2 shows the general architecture of each transformer network with the reconstructor module. As shown in 2, the student network takes the reference embeddings from the teacher network as well as the training sample as input. The teacher model only takes the training sample as input. To learn useful visual representations, DINO looks at two global crops and n local crops from the same image through the teacher and student model and compares their class token predictions. Since we are interested in not only useful semantic visual representations, but also semantic correspondences between image frames, we replace the two global crops from the same image with global crops from different frames from the same video. In our experiments we set δ to 1 and we do not use any local crops from either frames.

We follow the common vision transformer protocol of splitting the images into patches and add the positional embeddings. However, to discourage the model from cheating during the embedding reconstruction, we do not add the class token until after the reconstruction module. The patches go through one or more variable number of standard transformer blocks, then a reconstruction block that is used only in the student network.

We take the normalized softmax of the MLP head output. We use a temperature variable τ to normalize the logits, controlling the hardness or softness of the softmax output. We use the same notation as [1] where P_s and P_t are the student and teacher network's output probability distributions respectively, we denote P_s as:

$$P_s(x)^{(i)} = \frac{\exp(g_{\theta_s}(x)^{(i)}/\tau_s)}{\sum_{k=1}^K \exp(g_{\theta_s}(x)^{(k)}/\tau_s)}$$

The teacher's output, P_t follows the same formulation. The prediction loss which is the cross entropy loss is given by:

$$L_{pred} = H(P_t(x), P_s(x))$$

Where θ_s represents the student network's weights. The total loss is the sum of the reconstruction loss and the prediction loss and is given as:

$$L = L_{pred} + \lambda L_{recon}$$

where λ is the reconstruction loss weight that controls how much the reconstruction loss contributes to the total loss. We experiment with different λ values and record our observations in the experiments section. The student network learns to match the teacher network's output by minimizing the prediction loss w.r.t its weights, θ_s . Only the student network's weights are updated with the gradients from the loss. We do not calculate the gradients w.r.t the teacher network's weights. Instead, we use momentum update similar to [1] and update the teacher network's weights from the

student network's weights. The weight update for the teacher network is given as:

$$\theta_t \leftarrow m\theta_t + (1 - m)\theta_s$$

where θ_s is the teacher weight parameters, $m \in [0, 1]$ is the momentum coefficient and θ_s the student weight parameters.

IV. EXPERIMENTS

We use the same hyperparameters as DINO in our experiments. We try different positions for the reconstruction block - 1, 6 and 11 which correspond to having it right after the first, the 6th (middle) and the 11th (second to last) transformer blocks. In our preliminary experiments, we find that the loss goes down significantly faster when the reconstruction block was at index 1, that is right after the first transformer block. We notice that it deteriorates after the first few iterations and quickly goes back up. At index 6 the loss also goes down significantly at first, but the learning is not sustained. Our intuition is that the model is finding a trivial solution early in the training process and adjusting its weights towards that direction and gets stuck in a sub-optimal local maxima.

We train the model on the Kinetics-400 dataset [41], a dataset of curated videos of different human actions. We sample video frames at 10 frames per second. We run experiments with different λ values and observed that the reconstruction loss does not significantly affect the total loss. The prediction loss is the major contributor so the most impactful values for λ are 0 and 1. Throughout all of our experiments so we set p to 0.25. In future experiments we plan on trying different p values like 0.50, 0.75 and 100. We run the training for 100 epochs.

V. CONCLUSIONS AND FUTURE WORK

This is still a work in progress. While we are confident that our model will learn useful intra-frame semantic visual representations and inter-frame correspondences, we do not yet to have results to show. Our model is set up to learn intra-frame features and also inter-frame correspondences. We plan on running more experiments and adjusting the hyperparameters as needed to help the model learn. We also plan on adding code to visualize the attention maps to see what and how the model is learning.

ACKNOWLEDGMENT

This work was funded by Carnegie Mellon University through the Robotics Institute Summer Scholar program.

REFERENCES

- [1] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," *arXiv preprint arXiv:2104.14294*, 2021.
- [2] W. Wang, H. Song, S. Zhao, J. Shen, S. Zhao, S. C. Hoi, and H. Ling, "Learning unsupervised video object segmentation through visual attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3064–3074.
- [3] S. Caelles, J. Pont-Tuset, F. Perazzi, A. Montes, K.-K. Maninis, and L. V. Gool, "The 2019 davis challenge on vos: Unsupervised multi-object segmentation," 2019.
- [4] T. Zhuo, Z. Cheng, P. Zhang, Y. Wong, and M. Kankanhalli, "Unsupervised online video object segmentation with motion property understanding," *IEEE Transactions on Image Processing*, vol. 29, pp. 237–249, 2019.
- [5] H. Song, W. Wang, S. Zhao, J. Shen, and K.-M. Lam, "Pyramid dilated deeper convlstm for video salient object detection," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 715–731.
- [6] Z. Yang, Q. Wang, L. Bertinetto, W. Hu, S. Bai, and P. H. Torr, "Anchor diffusion for unsupervised video object segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 931–940.
- [7] T. Zhou, J. Li, X. Li, and L. Shao, "Target-aware object discovery and association for unsupervised video multi-object segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6985–6994.
- [8] S. Garg and V. Goel, "Mask selection and propagation for unsupervised video object segmentation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 1680–1690.
- [9] C. Yang, H. Lamdouar, E. Lu, A. Zisserman, and W. Xie, "Self-supervised video object segmentation by motion grouping," *arXiv preprint arXiv:2104.07658*, 2021.
- [10] S. D. Jain, B. Xiong, and K. Grauman, "Fusionseg: Learning to combine motion and appearance for fully automatic segmentation of generic objects in videos," 2017.
- [11] P. Tokmakov, K. Alahari, and C. Schmid, "Learning video object segmentation with visual memory," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4481–4490.
- [12] J. Luiten, I. E. Zulfiqar, and B. Leibe, "Unovost: Unsupervised offline video object segmentation and tracking," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 2000–2009.
- [13] T. Zhou, J. Li, S. Wang, R. Tao, and J. Shen, "Matnet: Motion-attentive transition network for zero-shot video object segmentation," *IEEE Transactions on Image Processing*, vol. 29, pp. 8326–8338, 2020.
- [14] W. Wang, J. Shen, X. Lu, S. C. Hoi, and H. Ling, "Paying attention to video object pattern understanding," *IEEE transactions on pattern analysis and machine intelligence*, 2020.
- [15] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2020.
- [16] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2794–2802.
- [17] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: unsupervised learning using temporal order verification," in *European Conference on Computer Vision*. Springer, 2016, pp. 527–544.
- [18] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, pp. 539–546 vol. 1.
- [19] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1422–1430.
- [20] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *ICLR 2018*, 2018.
- [21] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544.
- [22] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015. [Online]. Available: <http://arxiv.org/abs/1503.02531>
- [23] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Moment contrast for unsupervised visual representation learning," 2020.
- [24] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.
- [25] D. Liu, D. Yu, C. Wang, and P. Zhou, "F2net: Learning to focus on the foreground for unsupervised video object segmentation," 2020.
- [26] B. Duke, A. Ahmed, C. Wolf, P. Aarabi, and G. W. Taylor, "Sstvos:

- Sparse spatiotemporal transformers for video object segmentation,” 2021.
- [27] J. Mei, M. Wang, Y. Lin, and Y. Liu, “Transvos: Video object segmentation with transformers,” 2021.
- [28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008.
- [29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.
- [30] G. Bertasius, H. Wang, and L. Torresani, “Is space-time attention all you need for video understanding?” *arXiv preprint arXiv:2102.05095*, 2021.
- [31] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman, “Video action transformer network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 244–253.
- [32] S. Tulsiani and A. Gupta, “Pixeltransformer: Sample conditioned signal generation,” 2021.
- [33] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” in *European Conference on Computer Vision*. Springer, 2020, pp. 213–229.
- [34] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [35] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, “Vivit: A video vision transformer,” *arXiv preprint arXiv:2103.15691*, 2021.
- [36] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia, “End-to-end video instance segmentation with transformers,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 8741–8750.
- [37] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu, “Learning spatio-temporal transformer for visual tracking,” *arXiv preprint arXiv:2103.17154*, 2021.
- [38] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers & distillation through attention,” in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139, PMLR. PMLR, 18–24 Jul 2021, pp. 10 347–10 357. [Online]. Available: <http://proceedings.mlr.press/v139/touvron21a.html>
- [39] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilennets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.
- [40] X. Zhang, X. Zhou, M. Lin, and J. Sun, “Shufflenet: An extremely efficient convolutional neural network for mobile devices,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.
- [41] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, “The kinetics human action video dataset,” 2017.
- [42] G. O. Young, “Synthetic structure of industrial plastics,” in *Plastics*, 2nd ed., J. Peters, Ed. New York: McGraw-Hill, 1964, vol. 3, pp. 15–64.
- [43] K. Sofiuk, O. Barinova, and A. Konushin, “Adaptis: Adaptive instance selection network,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7355–7363.

Probabilistic Safe Reinforcement Learning using Control Barrier Function for Autonomous Vehicle Ramp Merging Control

Quanzhi Fu¹, Yiwei Lyu² and John M. Dolan³

Abstract—The ramp merging problem is one of the challenges for autonomous vehicles, since this is a situation where autonomous driving agents interact with human drivers. Learning a strategy to interact with humans is required for a safe merge. Reinforcement Learning (RL) is an end-to-end learning method for policy finding. Control barrier functions (CBF) are commonly used to achieve a forward invariance safety guarantee. RL can be combined with CBF to acquire a safe merge policy. Current works use rollout to ensure the output action is within the safety constraints. A problem with existing rollout algorithms is that they are based on the assumption that the exact dynamics of the environment are known and the future states can be exactly predicted. However, due to the inherent uncertainty, in reality, the future states are not fully predictable. In this work, instead of penalizing the rollout states that violate the safety constraints, we constrain the output action to be inside an action set that guarantees a given safety level. The performance of the proposed Safety-Assured Policy Optimization for Ramp Merging (SAPO-RM) algorithm is compared with Constrained Policy Optimization (CPO). SAPO-RM updates the policy safely with a safe initial policy. However, in the contrast with CPO, it fails to make an unsafe policy safe through the update. Moreover, SAPO-RM sometimes shows a sudden increase in return, which benefits the convergence of the algorithm.

I. INTRODUCTION

Safety is critical for autonomous driving. Collisions must be avoided in all circumstances to prevent harm to humans. Existing autonomous driving algorithms can control the car safely in most scenarios. However, the autonomous vehicle still faces risks when it needs to directly interact with human-driven vehicles, as people have diverse driving styles. A typical traffic scenario where an autonomous driving vehicle needs to interact with human-driven vehicles is ramp merging (see Fig. 1).

Control barrier functions (CBF) can provide an admissible control space for certain safety constraints in a dynamical system. The forward invariance property of CBF gives a strong guarantee of safety. Compared with traditional pointwise constraints, CBF enforces a more cautious approach to the constraint boundary [2] (see Fig. 2).

Reinforcement learning (RL), as an end-to-end learning method for sequential decision making, has shown powerful

¹The author is with the School of Data Science, the Chinese University of Hong Kong, Shenzhen, Guangdong, 518172 China. This work is completed when Quanzhi Fu served as an intern in the Robotics Institute Summer Scholar, Carnegie Mellon University. Email: quanzhifu@link.cuhk.edu.cn

²The authors are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, 15213 USA. Email: yiweilyu@andrew.cmu.edu

³The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA. Email: jmd@cs.cmu.edu

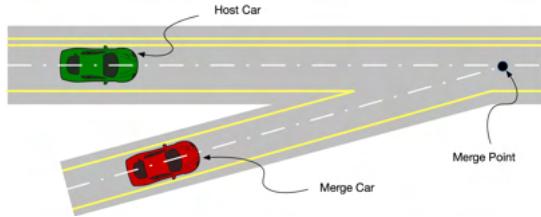
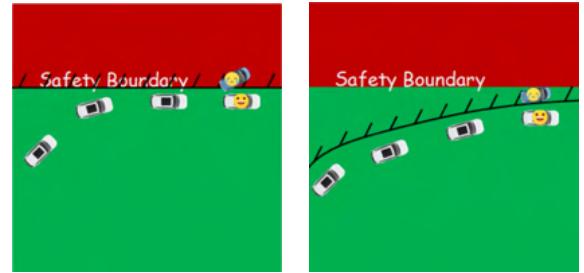


Fig. 1: Merge scenario [1]. The host car (green) is an autonomous vehicle, running on the main road; the merge car (red) is a human driven car, running on the ramp.



(a) Traditional Pointwise Constraint (b) Control Barrier Function

Fig. 2: Intuitive depiction of control barrier functions.

performance in a lot of control problems [3], [4] and has also been widely applied in the autonomous driving field. A safety-concerned RL problem can be formulated as a constrained RL problem in which agents learn to maximize the expected return while always satisfying the safety constraints. Existing constrained RL algorithms always use inequalities to bound cost functions as the constraints. Constrained policy optimization (CPO) is an algorithm that provides a theoretical guarantee for safe exploration [5]. It applies the accumulated cost as the constraint and also uses a trust-region constraint for stable updating. Trust-region [6] is a method proposed to confine the update step length of the policy to balance the sample efficiency and stability of the learning. With a linear approximation technique, the CPO algorithm can be conducted very efficiently while generally retaining safety. Therefore, CPO is one of the most commonly used baselines for safe-RL.

Although the existing constrained RL algorithms can satisfy the safety constraints for policy updates, the constraints are bounds of the expected cost. Satisfying the constraints does not provide an absolute safety guarantee for the policy. Thus the update direction for the given constrained policy optimization problem cannot ensure safety. A way to tackle

this problem is to adopt the model information into the algorithm [7]. In [2], the authors propose a method that uses a CBF constraint to replace the accumulated cost. The forward invariance property of the CBF provides a theoretical guarantee for safe exploration.

Adopting model information into the algorithm does solve the problem. A reason why interacting with the human-driven car is especially challenging for autonomous vehicles is that humans introduce uncertainty into the merging process. In other words, future states are a random distribution condition on current state and action. Algorithms with model information require the use of roll-out to predict the future states. However, due to the uncertainty in the system dynamics, the roll-out faces a severe variance inflation problem. This problem harms the performance of model-based algorithms in the system whose dynamics contain uncertainty.

In this paper, we propose a probabilistic safety-assured RL algorithm Safety-Assured Policy Optimization for Ramp Merging (SAPO-RM) for the ramp merging problem. We adopt a constraint on output actions that provides a probabilistic safety guarantee on the dynamics with uncertainty in our algorithm. SAPO-RM is in actor-critic framework. Both the actor and the critic are approximated by artificial neural networks (ANN) and a linear approximation technique is used for efficient updating. Moreover, we add a trust-region constraint to ensure a monotonic updating of the policy.

The rest of the paper is organized as follows. In section II we summarize some recent work in constrained RL. In section III-D, we provide some preliminaries about constrained RL and control barrier functions. The problem formulation is also introduced in this section. In section IV we elaborate on the details of the proposed algorithm. Section V-B contains the simulated result and the analysis. Section VI provides some future directions for this work.

II. RELATED WORK

Constrained RL was introduced to address the problem of training RL agents with constraints. The goal for constrained RL is to achieve a safe exploration during the training and exploit phase. The constrained RL agent aims to find a policy π that maximizes the long-term return G .

$$\pi^* = \arg \max_{\pi} \mathbb{E}_{a_t \sim \pi}[G = \sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t)] \quad (1)$$

where $r(\cdot)$ denotes the reward function and s_t, a_t denote state and action respectively.

Constrained RL agents are designed to optimize the objective in the feasible region Π_C .

$$\Pi_C = \{J_c \leq d_c\} \quad (2)$$

where J_c is the cost function for constraint c and d_c is the selected threshold for c .

One method of combining CBF with RL is to design a cost function for the CBF and add it into the constraints. In addition to the cost function, constraint formulation has a critical effect on constrained RL algorithms. A very early intuitive

way to formulate the constraint is Policy Gradient Projection (PGP), which uses the average cost as the constraint [8]:

$$\lim_{T \rightarrow \infty} \left[\mathbb{E}_{s \sim d(s), a \sim \pi_k} \left(\frac{1}{T} \sum_{t=1}^T r_{C_i} \right) \right] \leq d_i \quad (3)$$

where r_{C_i} is a one-hot vector containing the constraint violation cost. This formulation is intuitively reasonable, but it always allows some constraint violations, as it is based on average cost. To tackle this problem, the authors of [9] adopt a balance parameter η in the constraint formulation.

$$\lim_{v \in \mathbb{R}} \left\{ v + \frac{1}{1-\eta} \mathbb{E}_{s \sim d(s), a \sim \pi_k} [(r_{C_i} - v)^+] \right\} \leq d_i \quad (4)$$

This method alleviates the problem caused by some low-probability actions having severe bad consequences. However, the nature of the balancing parameter determines that low-probability violation is still accepted.

CPO is the first algorithm that claims to guarantee safe exploration. The updating of CPO is based on reward advantage and cost advantage. CPO also adopts a trust region for faster and stabler policy updating [5]:

$$\begin{aligned} \pi_{k+1} &= \arg \min_{\pi} \mathbb{E}_{s \sim d^{\pi_k}, a \sim \pi_k} [A^{\pi^k}(s, a)] \\ \text{s.t. } J_{C_i}(\pi_k) + \frac{1}{1-\gamma} \mathbb{E}_{s \sim d^{\pi_k}, a \sim \pi_k} [A_{C_i}^{\pi^k}(s, a)] &\leq d_i \\ \overline{D}_{KL}(\pi || \pi_k) &\leq \delta \end{aligned} \quad (5)$$

where $A^\pi(s, a)$ denotes the advantage function, which is defined as the difference between state-action function Q and state-value function V :

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s) \quad (6)$$

With linear approximation techniques, CPO can be performed very efficiently. Thus it is a commonly used baseline for constrained RL. While it accept constraint violations for the system with randomness.

Another method to combine CBF and constrained RL is using a hierarchy architecture. The feasibility for the outputs of the RL agent is checked and modified by a CBF controller, thus ensuring that the output is always feasible [10]. In [10], the authors use CBF and CLF to constrain the output of the RL agent, thus achieving safe exploration. They also use Gaussian Processes to learn the model information of the environment, thus enabling model-based learning. Such methods use online learning, leading to the problem of low sample efficiency, and will therefore not be the focus of this paper.

In this paper, instead of use constraints of cost, we apply a CBF-based probabilistic safety assured constraint to the constrained optimization problem for the actor. Therefore, provide a update rule with theoretical safety guarantee.

III. PRELIMINARIES AND PROBLEM FORMULATION

A. Constrained reinforcement learning

RL is an algorithm for sequential decision-making control problems. In a general RL framework [11], an agent interacts

with an environment and learns from the feedback in a sequence of discrete timesteps to accomplish a task. The environment that the agent interacts with must have the Markov property, which means the future states only depend on the current state and the current input.

A commonly used RL architecture is actor-critic. Both actor and critic are function approximators, e.g. ANN, to approximate the policy $\pi(s; \theta)$ and value function of the policy $V^\pi(s; w)$, respectively.

The training process for the actor-critic RL algorithm contains two interactive steps: (1) evaluate the value function for the underlying policy, which is called policy evaluation, and (2) update the current policy based on the estimated value function given by the critic, which is called policy improvement.

The aim for constrained RL is to return a policy that maximizes the accumulated reward $G = \sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t)$ while satisfying the constraints. For the policy evaluation process, the critic is updated by iteratively minimizing the loss function,

$$L(w) = \mathbb{E}_{s \sim d} \left\{ \frac{1}{2} (r(s, a) + V^\pi(s'; w) - V^\pi(s; w))^2 \right\} \quad (7)$$

where d is the stationary distribution for the current policy and s' is the state after taking action a at state s .

For the policy improvement step, the parameter θ is optimized to maximize the estimated accumulated reward $\hat{G} = r(s, a) + V^\pi(s'; w)$.

$$\begin{aligned} & \max_{\theta} \mathbb{E}_{s \sim d, a \sim \pi(s; \theta)} [\hat{G}] \\ & \text{s. t. } g(s, a) \leq 0 \\ & \quad t(x, a) = 0 \end{aligned} \quad (8)$$

By alternately solving equations 7 and 8, a local optimal policy that satisfies the constraints can be obtained. Note that, since in equation 8 we are solving a constrained optimization problem, there may be no feasible solutions. Achiam et al. adopted a retrieval mechanism for policy updating to tackle this problem [5]. This mechanism will be elaborated in section IV.

B. Control Barrier Function

CBF is proposed to address safety with dynamic systems [12]. For a nonlinear system in a control affine form,

$$\dot{s} = f(s) + g(s)u \quad (9)$$

where $s \in \mathbb{R}^n, u \in \mathbb{R}^m$ are the system state and control input, respectively. For a function $h : \mathbb{R}^n \rightarrow \mathbb{R}$, define the safety space

$$\mathcal{H} = \{h(x) \geq 0\} \quad (10)$$

Then the definition of CBF for the safety set \mathcal{H} is as follows [12].

Definition 1. (Control Barrier Function) Given a dynamical system 9 and the set \mathcal{H} defined in 10 with a continuously differentiable function $h : \mathbb{R}^n \rightarrow \mathbb{R}$, then h is a control

barrier function (CBF) if there exists an extended class \mathcal{K}_∞ function for all $s \in S$ such that

$$\sup_{u \in \mathcal{U}} L_f h(s) + L_g h(s)u \geq -\kappa(h(s)) \quad (11)$$

where $S \subseteq \mathbb{R}^n, \mathcal{U} \subseteq \mathbb{R}^m$ denote the state space and action space of the system, respectively. $L_f h(s), L_g h(s)$ denote the Lie derivatives of h along vector field f and g .

Consistent with [13], we particularly choose a particular \mathcal{K}_∞ function, i.e., $\kappa(h(s)) = \alpha h(s)$, where α is the conservativeness coefficient.

Since RL is only compatible with a discrete-time environment, we adopt the discrete-time version of the CBF [2].

Definition 2. . (Discrete Time Control Barrier Function) The discrete-time control barrier function (CBF) for a constraint $h(s_t) \geq 0$ is

$$h(s_{t+1}) \geq (1 - \alpha)h(s_t) \quad (12)$$

In this definition, a difference replaces the derivative in the continuous situation. Thus, the admissible space in equation 11 is redefined as:

$$\mathcal{B}(u_t) = \{u : h(s_{t+1}|s_t, u_t) - h(s_t) \geq \alpha h(s_t)\} \quad (13)$$

It is proved that, as long as $s_0 \in \mathcal{H}$ and all subsequent actions $u_i \in \mathcal{B}(u_i), \forall i > 0$, the state s_i is always in the safe set \mathcal{H} : $s_i \in \mathcal{H} \forall i > 0$ [12]. This property is also known as the forward invariance property for CBF.

In this paper, we choose the same pairwise safety function h as [13]. The safety set \mathcal{H} and admissible set $\mathcal{B}(s)$ are redefined as

$$\begin{aligned} \mathcal{H} &= \{h(s) = \|x_e - x_h\|^2 - R_{safe}^2 \geq 0\} \\ \mathcal{B}(s_t) &= \{u_t : h(s_{t+1}|s_t, u_t) - h(s_t) \geq \alpha h(s_t)\} \end{aligned} \quad (14)$$

where x_e, x_h denote the position of the ego vehicle and the host vehicle and R_{safe} denotes the safe distance.

C. Problem Formulation

We consider the ramp merging scenario with Gaussian random noise [13]. Our agents are trained to control the ego vehicle on the main road to merge with a human-driven host vehicle from the ramp safely. Meanwhile, maintain the vehicle under the expected acceleration as much as possible. The human-driven vehicle is assumed to have constant velocity with some random noise. The system dynamics for each vehicle can be described as a double integrator as follows [13]:

$$\dot{X} = \begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0_{2 \times 2} & \mathbb{I}_{2 \times 2} \\ 0_{2 \times 2} & 0_{2 \times 2} \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} \mathbb{I}_{2 \times 2} & \mathbb{I}_{2 \times 2} \\ 0_{2 \times 2} & 0_{2 \times 2} \end{bmatrix} \begin{bmatrix} u \\ \epsilon \end{bmatrix} \quad (15)$$

where $\mathbb{I}_{2 \times 2}$ denotes a 2×2 identity matrix, and $x, v \in \mathbb{R}^2$ are the position and velocity of the vehicle. $u \in \mathbb{R}^2$ is the control input acceleration, and $\epsilon \sim \mathcal{N}(\mu, \Sigma) \in \mathbb{R}^2$ is the uncertainty in the system with known mean μ and covariance matrix Σ .

In this paper, we consider the discrete-time system dynamics:

$$X_{t+1} = \begin{bmatrix} x_{t+1} \\ v_{t+1} \end{bmatrix} = \begin{bmatrix} x_t + v_t \Delta t + \frac{1}{2} u_t \Delta t^2 \\ v_t + u_t \Delta t \end{bmatrix} \quad (16)$$

where Δt is the time increment. The goal for our algorithm is to achieve a safe policy while maintaining efficiency: i.e., satisfy the constraints and ensure that the acceleration does not deviate from the expectation too much. Thus we define the reward function r as

$$r(s_t, a_t) = \|a_t - a_{\text{expected}}\|^2 \quad (17)$$

where a_{expected} is selected by the user. The dynamics of the considered ramp merging problem contain randomness, thus we cannot fully guarantee the CBF constraints. Instead, we consider satisfying a constraint with a certain confidence level $\eta \in (0, 1)$. The desired parameter θ^* for the policy is the solution of the following constrained optimization problem:

$$\begin{aligned} & \min_{\theta} \mathbb{E}_{s \sim d, a \sim \pi(s; \theta)} [G] \\ \text{s.t. } & \Pr(\mathbb{E}_{a \sim \pi(s_t; \theta)} [h(s_{t+1} | s_t, a_t)] \geq (1 - \alpha)h(s_t)) \geq \eta \end{aligned} \quad (18)$$

D. Admissible set for probabilistic ramp merging system

Generally, optimization problems containing probabilistic constraints are hard to optimize. However, for the ramp merging setting we considered, the probabilistic constraint can be converted into a constraint on the output action [13].

Theorem 3. Given a stochastic dynamical system defined in equation 16 and a confidence level $\eta \in (0, 1)$, the following admissible control space $\mathcal{B}_\eta^s(x)$ ensures a chance-constrained safety condition $\Pr(\mathbb{E}_{a \sim \pi(\theta)} [h(s_{t+1})] \geq (1 - \alpha)h(s_t)) \geq \eta$ for the ego vehicle with each merging car m .

$$\mathcal{B}_\eta^s(x) = \{u_e \in \mathcal{U}_e : A_{em}u_e \leq b_{em}, \forall m\} \quad (19)$$

with

$$\begin{aligned} A_{em} &= -2\Delta x_{em}^\top \Delta t \\ b_{em} &= 2\Delta x_{em}^\top (\Delta v_{em} + \Delta \epsilon_{em}) + \alpha h_{em}^s(x) \\ &\quad - \Phi^{-1}(\eta) \sqrt{\Delta x_{em}^\top \Delta \Sigma_{em} \Delta x_{em}} \end{aligned} \quad (20)$$

where $\Delta x_{em} = x_e - x_m$, $\Delta v_{em} = v_e - v_m$, $\epsilon_{em} = \epsilon_e - \epsilon_m \sim N(\Delta \hat{\epsilon}_{em}, \Delta \Sigma_{em})$ for ego vehicle e and each merging vehicle m .

The detailed proof can be found in [13].

IV. METHODS

The aim for reinforcement learning algorithms is to maximize the expected return G . We use a policy-based reinforcement learning algorithm. To guarantee a safe merge, updating the policy is a constrained optimization problem:

$$\begin{aligned} & \min_{\Delta \theta} L_a = \mathbb{E}_{s \sim \mathcal{C}, a \sim \pi(\theta)} [G] \\ \text{s.t. } & J_c = \mathbb{E}_{u_e \sim \pi(\theta)} [A_{em}u_e] \leq b_{em} \\ & \bar{D}_p(\theta; \theta_k) \approx \frac{1}{2}\Delta \theta^\top H \Delta \theta \leq \delta \end{aligned} \quad (21)$$

with

$$\begin{aligned} A_{em} &= -2\Delta x_{em}^\top \Delta t \\ b_{em} &= 2\Delta x_{em}^\top (\Delta v_{em} + \Delta \hat{\epsilon}_{em}) + \alpha h_{em}^s(x) \\ &\quad - \Phi^{-1}(\eta) \sqrt{\Delta x_{em}^\top \Delta \Sigma_{em} \Delta x_{em}} \end{aligned} \quad (22)$$

The updating of the critic is the same as for the constraint-free actor-critic algorithm, by optimizing the critic loss defined in equation 23:

$$L_w = \mathbb{E}_{s_t \sim \mathcal{C}} \left\{ \frac{1}{2}(G - V(s_t; w))^2 \right\} \quad (23)$$

analogously to the method used in [2]. An approximate solution of the constrained optimization can be computed through linear approximation and a trust region constraint is added for a more stable convergence:

$$\begin{aligned} & \min_{\Delta \theta} g^\top \Delta \theta \\ \text{s.t. } & z + C^\top \Delta \theta \leq 0 \\ & \bar{D}_p(\theta; \theta_k) \approx \frac{1}{2}\Delta \theta^\top H \Delta \theta \leq \delta \end{aligned} \quad (24)$$

where $g = \frac{\partial L_a}{\partial \theta} / \| \frac{\partial L_a}{\partial \theta} \|^2$, $z = J_c - b_{em}$, $C = \frac{\partial J_c}{\partial \theta} / \| \frac{\partial J_c}{\partial \theta} \|^2$.

Using a Lagrange multiplier, the Lagrangian function is

$$L(\Delta \theta, v) = g^\top \Delta \theta + \lambda \left(\frac{1}{2}\Delta \theta^\top H \Delta \theta - \delta \right) + v(z + C^\top \Delta \theta) \quad (25)$$

where λ and v are dual variables. Using the KKT condition,

$$\begin{aligned} \frac{\partial L}{\partial \Delta \theta} &= g + \lambda H \Delta \theta + v C = 0 \\ \lambda \left(\frac{1}{2}\Delta \theta^\top H \Delta \theta - \delta \right) &= 0 \\ v(z + C^\top \Delta \theta) &= 0 \\ \lambda, v &\geq 0 \\ \frac{1}{2}\Delta \theta^\top H \Delta \theta - \delta &\leq 0 \\ z + C^\top \Delta \theta &\leq 0 \end{aligned} \quad (26)$$

the optimal update direction can be obtained. Supposing there exists a feasible solution for the optimization problem 24, the optimal update direction is

$$\Delta \theta = \frac{H^{-1}(g - v^* C)}{\lambda^*} \quad (27)$$

where v^* and λ^* are the optimal dual solution obtained by analytical solution. If the problem does not have a feasible solution the policy update rule changes to a retrieval mechanism:

$$\theta_{k+1} = \theta_k - \sqrt{\frac{2\delta}{C^\top H^{-1} C}} H^{-1} C \quad (28)$$

For the retrieval update, we tentatively ignore the objective function and take the gradient descent with respect to the constraints; in other words, we try to "force" the policy back to the safety region.

Algorithm 1 SAPO-RM

input: arbitrary initialized parameter θ_0, w_0
output: trained parameters θ^*, w^*

repeat

- Sample a set of trajectories $\mathcal{D} = \mathcal{T} \sim \pi(\theta_k)$
- From \mathcal{D} update w by (23)
- Estimate g, z, C, H
- Check the feasibility by (30)
- if** feasible **then**

 - update θ by (27)

- else**

 - update θ by (28)

- end if**

until converge

input
y_e The y-position for the ego vehicle
x_h The x-position for the host vehicle
y_h The y-position for the host vehicle
v_e The velocity for the ego vehicle
v_h The velocity for the host vehicle
output
a The acceleration of the ego vehicle

TABLE I: summary of the input and output for the RL information

We check the feasibility by solving the following optimization problem, which is proposed in [7]:

$$\begin{aligned} \min_{\Delta\theta} \quad & \frac{1}{2} \Delta\theta^T H \Delta\theta \\ \text{s.t. } & z + C^\top \Delta\theta \leq 0 \end{aligned} \quad (29)$$

Suppose the optimal value of equation 29 is δ_{min} . The feasible solution set is empty if $\delta_{min} \geq \delta$ and contains a feasible solution otherwise. We optimize this problem efficiently using the Lagrangian dual problem:

$$\max_{v \geq 0} -\frac{-v^\top C^\top H^{-1} Cv}{2} + v^\top z \quad (30)$$

The feasibility check can be done by comparing δ_{min} with δ .

The probabilistic Safety Assured Policy Optimization for Ramp Merging (SAPO-RM) is summarized in Alg. 1.

V. EXPERIMENTAL RESULTS

We conduct experiments to test the sample efficiency and the ability to perform safe exploration. We use a simulated environment for ramp merging which contains one ego vehicle and one host vehicle. The RL agent is trained to safely merge with the host vehicle that comes from the ramp. The host vehicle has a constant velocity with random variance. The observation of the RL agent $s = (y_e, x_h, y_h, v_e, v_h) \in \mathbb{R}^5$, and the control input for the ego vehicle is the acceleration a . The basic information for the RL algorithm is summarized in Table I. The reward function is defined in equation 17.

The environment dynamics follow equation 16. We assume a speed limit $v_{max} = 35m/s$ for the environment. The ego vehicle is controlled not to exceed this speed limit.

The architectures of the neural networks used in the experiments are summarized in Table II. The optimizer for the

Actor	
Input Layer	5->120 fully connected layer
Activation	Tanh
Hidden Layer	120->1 fully connected layer
Activation	Tanh
Output Layer	1-d Gaussian Layer
	After the second activation function, the value are multiplied by a constant to confine the output action between the feasible region
Critic	
Input Layer	5->80 fully connected layer
Activation	Tanh
Output Layer	80->1 fully connected layer

TABLE II: Summary of the neural networks used in the experiments

critic network is LBFGS. The gradient of $\mathbb{E}_{u_e \sim \pi(\theta)}[A_{em} u_e]$ is estimated using the REINFORCE trick,

$$\begin{aligned} \nabla J_c &= \int \nabla p(u_e; \theta) \cdot A_{em} \cdot u_e du_e \\ &= \int p(u_e) \nabla \ln p(u_e; \theta) \cdot A_{em} \cdot u_e du_e \\ &= \nabla \mathbb{E}_{u_e \sim \pi(\theta)}[\ln p(u_e; \theta) \cdot A_{em} \cdot u_e] \end{aligned} \quad (31)$$

Multiple actions u were sampled from the policy distribution and timed with the log probability. The mean of this value is an unbiased estimator for the constraint loss.

For each experiment, we start with fixed initial parameters for both algorithms to fairly compare their performance.

Note that due to time limitations, the experimental results shown in the following sections are based on very limited testing. More systematic experiments will be performed in the future.

A. Experiment 1: safe exploration

In this experiment, we aim to answer the following question: Does our algorithm provide more safety in a ramp merging environment with uncertainty?

In this experiment, we chose episode constraint violation distance to evaluate constraints violation for each algorithm, which is defined in equation 32:

$$\mathbb{E}_{\mathcal{T}}[R_{safe}^2 - \|x_e - x_h\|^2]^+ \quad (32)$$

where \mathcal{T} denotes trajectories and $[\cdot]^+$ denotes the function $\max\{\cdot, 0\}$. We use this as the evaluation criterion for safety constraints, since we not only care about the frequency of constraint violation, but also the extent of violation. Policies with less frequent but severe violations are not what we want. Thus we use episode constraint violation distance to capture both violation frequency and the extent of the violation. The smaller the constraints violation distance is, the better the feasibility performance algorithm. The performance of algorithms during the training process is shown in figure 3.

Compared with CPO, SAPO-RM shows greater variance when dealing with constraint violations. The CPO generally monotonically reduces the violation rate. SAPO-RM, in the contrast, shows a huge fluctuation in violation distances. In some other trials of the experiment, SAPO-RM fails to reduce the violation even after thousands of epochs. This indicate that there is no guarantee for SAPO-RM to get a safe policy when the initial policy is unsafe. A possible reason for

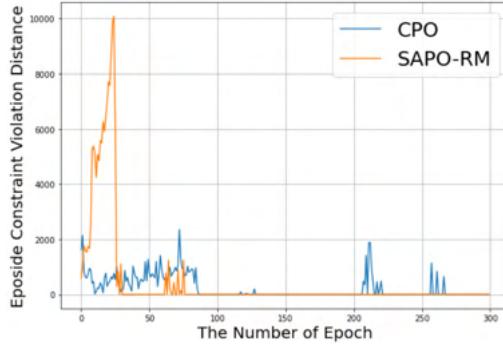


Fig. 3: Average episode constraints violation distance with different algorithms.

this is that SAPO-RM is constrained by the mean of the CBF-based step-wise constraints. CBF-based constraints have a safety guarantee only when the initial policy π_0 is a feasible policy. However, there is no guarantee for the update if the agent is initially not in a safe state. Another possible reason is due to the constraint being step-wise. Taking the mean of these inequalities over trajectories cannot guarantee that all constraints are satisfied, since these constraints may conflict with each other. Thus, we propose an updated version of SAPO-RM which conducts step-wise optimization (see Alg. 2). We will test its performance in the future. The current version of SAPO-RM has worse performance in dealing with violations compared with CPO, as it fails to provide a stable update that makes the policy go back to the safe region. However, once SAPO-RM has a generally safe policy, no matter a safe initial policy or a occasionally gotten safe policy during the training process, it can update the policy to remain in the feasible region.

B. Experiment 2: sample efficiency

In this experiment, we will test the algorithm's episode return, to see the sample efficiency. Sample efficiency is a critical problem for RL algorithms. RL agents learn from interaction with the environment, which is costly in reality. We test the average return for the CPO and SAPO-RM algorithms. The results are shown in Figure 4.

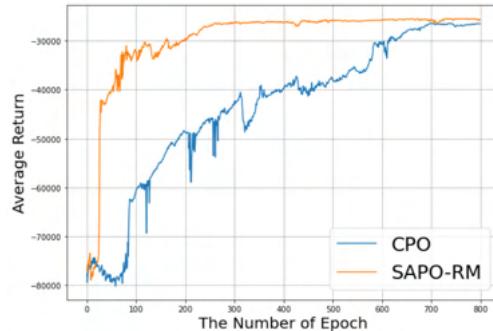


Fig. 4: Average episode return with different algorithms.

CPO shows an erratic reward curve, as it doesn't get a safe policy until about 300 epochs. The shape of the reward

Algorithm 2 SAPO-RM Updated

input: arbitrary initialized parameter θ_0 , w_0
output: trained parameters θ^* , w^*

```

repeat
    Sample a set of trajectories  $\mathcal{D} = \mathcal{T} \sim \pi(\theta_k)$ 
    From  $\mathcal{D}$  update  $w$  by (23)
    for  $t = 1 : T$  do
        Estimate  $g, z, C, H$  by  $s_t$  and  $a_t$ 
        Check the feasibility by (30)
        if feasible then
            update  $\theta$  by (27)
        else
            update  $\theta$  by (28)
        end if
    end for
until converge

```

curve greatly depends on the initialization of the policy. More experiments are required to explore the effect of the initialization. For initialization where CPO can reach a safe policy in a few steps, the reward curve becomes smooth. A special observation for SAPO-RM is that once SAPO-RM reaches a safe policy, its reward makes a huge jump. This jump occurs frequently in the training and significantly accelerates the convergence of the algorithm. More experiments are required to delve into the reason for this jump.

VI. FUTURE WORK

This paper only contains some preliminary experiments with limited data. More experiments are required for a comprehensive analysis of SAPO-RM. The performance of Updated SAPO-RM (Alg. 2) is worth testing, since it may be able to deal with an unsafe initial policy.

VII. ACKNOWLEDGMENT

This work is supported by the CMU Robotics Institute Summer Scholars (RISS) program. This work also received financial support from the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS) and the Chinese University of Hong Kong, Shenzhen.

The author (Quanzhi Fu) would like to thanks Spencer Van Koevering, Yiwei Lyu, Wenhao Luo, and Professor John Dolan for their valuable suggestions. The author also wants to give special thanks to his family, his girl friend Qiyu, friends Zhanghao, and Ang, who gave a lot of support when the author was under great pressure.

REFERENCES

- [1] C. Dong, J. M. Dolan, and B. Litkouhi, "Intention estimation for ramp merging control in autonomous driving," in *2017 IEEE intelligent vehicles symposium (IV)*. IEEE, 2017, pp. 1584–1589.
- [2] H. Ma, J. Chen, S. E. Li, Z. Lin, Y. Guan, Y. Ren, and S. Zheng, "Model-based constrained reinforcement learning using generalized control barrier function," *arXiv preprint arXiv:2103.01556*, 2021.
- [3] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [4] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *International conference on machine learning*. PMLR, 2014, pp. 387–395.
- [5] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International Conference on Machine Learning*. PMLR, 2017, pp. 22–31.

- [6] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.
- [7] J. Duan, Z. Liu, S. E. Li, Q. Sun, Z. Jia, and B. Cheng, “Deep adaptive dynamic programming for nonaffine nonlinear optimal control problem with state constraints,” *arXiv preprint arXiv:1911.11397*, 2019.
- [8] E. Uchibe and K. Doya, “Constrained reinforcement learning from intrinsic and extrinsic rewards,” in *2007 IEEE 6th International Conference on Development and Learning*. IEEE, 2007, pp. 163–168.
- [9] Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone, “Risk-constrained reinforcement learning with percentile risk criteria,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6070–6120, 2017.
- [10] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, “End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 3387–3395.
- [11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [12] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3420–3431.
- [13] Y. Lyu, W. Luo, and J. M. Dolan, “Probabilistic safety-assured adaptive merging control for autonomous vehicles,” *arXiv preprint arXiv:2104.14159*, 2021.

Automatic Multi-modal Calibration of Stereo Cameras, Thermal Cameras, and Lasers in Arbitrary Scenes

Taimeng Fu¹, Huai Yu², Yaoyu Hu² and Sebastian Scherer²

Abstract—The combined use of multi-modal sensors has become increasingly popular in autonomous driving and intelligent robots, since the integration can provide richer information than any single sensor, enhance the reliability under complex environments. The precise 6-DoF extrinsic calibration between any two sensors of a perception system is important since it directly affects its performance. Traditional calibration methods are only designed for calibrating sensors in no more than two modalities, and most of them either require specific calibration targets or need manual manipulation. It's inconvenient to use them as the sensor types increase.

In this paper, we introduce an automatic cross-modal calibration framework, which can calibrate the extrinsic parameters between stereo cameras, thermal cameras, and LiDAR sensors in one shot. It can be conducted in arbitrary scenes without any specific calibration target, and perform automatically without manual intervention. The key insight is to extract edge information from the data of different sensors and align these edges by minimizing the registration error on extrinsic parameters. Although the cost function is not convex globally, it is always convex locally around the correct calibration. The algorithm only needs a rough initialization, which can be manually obtained, and then it will optimize this initial guess to estimate the precise extrinsic parameters.

I. INTRODUCTION

Today, robots have been used to automatically perform a variety of challenging tasks, and perception of the environment is a critical step in accomplishing them. To maximally collect environmental information, the latest robots are often equipped with different types of sensors, such as stereo cameras, thermal cameras, and LiDAR sensors. Extrinsic calibration is a process of estimating the rigid-body transformation between each two sensors' local coordinates. With the transformation, 3D points detected by different sensors can be converted to a unified coordinate system or projected onto camera images. As the sensor types grows, accurate extrinsic calibration becomes increasingly important in multi-modal perception systems. Calibration serves as a bridge connecting sensors, providing fusion information for the perception system, and enhancing the robot's ability to perceive the environment.

Over the past several years, substantial works have been done on cross-modal extrinsic calibration. It is a challenging problem since it's difficult to perform automatic feature matching between different modalities. We will introduce

¹Taimeng Fu is with the School of Data Science, The Chinese University of Hong Kong, Shenzhen, 2001 Longxiang Boulevard, Longgang District, Shenzhen, China, 518712 taimengfu@link.cuhk.edu.cn

²Huai Yu, Yaoyu Hu and Sebastian Scherer are with the AirLab, Carnegie Mellon University, Pittsburgh, PA 15213 {huaiy, yaoyuh, basti}@andrew.cmu.edu

previous works on calibrating monocular camera and lasers, stereo cameras and lasers, and RGB and thermal cameras respectively.

A. Calibration of monocular camera and lasers

Zhang and Pless [1] proposed a method to calibrate the camera and lasers system with a checkerboard calibration target that can be simultaneously observed by both sensors. Later, Scaramuzza et al. [2] demonstrated an improved laser point cloud visualization technique that enables manually matching laser points with the corresponding image pixels. They employed the perspective-from-n-points (PnP) algorithm to calculate the transformation based on the matches. Their method does not depend on a specific calibration target, so it can be executed in any scene. However, manual matching is time-consuming and might be imprecise. On the other side, Nunez et al. [3] chose to increase the degree of automation while still under a controlled scene. They developed an algorithm to automatically detect the checkerboard in the lasers' view and align it to the camera image to get extrinsic parameters. To improve the accuracy, multiple laser frames were aggregated with the help of an inertial measurement unit (IMU).

There were also some attempts to get rid of both fixed scene and manual manipulation. Pandey et al. [4] provided a solution of automatic targetless extrinsic calibration by maximizing mutual information of laser reflectivity and image intensity. They found that after aggregating 10 scans, their MI cost function becomes convex and thus easy to optimize. However, the laser reflectivity and image intensity might not be strongly correlated in some scenarios. This might result fault calibration. Levinson and Thrun [5] developed an online system that automatically align laser edges to image edges to correct sensor drifting. They generated cost maps for each image, projected laser edge points onto them to calculate the cost, and tried to adjust the extrinsic parameters to lower the cost. However, on one hand, since the laser scans are relatively sparse, some of the laser edge points may not precisely lie on the boundary of the objects, which affects the calibration accuracy; on the other hand, they used a greedy approach to reduce the cost, which is less efficient than a gradient based optimizer.

B. Calibration of stereo cameras and lasers

The stereo-laser calibration is more straightforward than that of monocular camera and lasers since both stereo cameras and lasers have depth information. Guindel et al. [6] calibrated stereo cameras and lasers based on a four-hole

calibration board. They developed a segmentation pipeline for extracting boundaries of the holes from stereo point cloud and laser point cloud respectively, and calculated the extrinsic parameters by minimizing the distance between the clustered centroids of the four holes. Dhall et al. [7] reported a method that relies on a calibration board with a visual tag on it. They calculate the 3D position of the board corners in both stereo and lasers' view, and solve a set of equations to get the transformation that minimizes the distance between corresponding corner points.

C. Calibration of RGB and thermal cameras

In recent years, with the development and wide use of thermal cameras, RGB-thermal calibration becomes an unavoidable task for image fusion and object fusion. Most of the attempts tried to build calibration targets that are distinguishable in both RGB and thermal views. Li et al. [8] placed LED bulbs on their calibration board. The bulbs have high intensity on both RGB and thermal images since they emit both light and heat. They developed an algorithm to localize the bulbs on RGB and thermal images, and then calibrate the extrinsic parameters by minimizing the reprojection error. Shivakumar et al. [9] mounted aluminum squares on a black acrylic background to form a checkerboard. Since the checkerboard blocks have different colors and different thermal reflectivities, it's easy to recognize the chessboard pattern on both RGB and thermal images. They then employed OpenCV's [10] camera calibration toolbox to estimate the transformation.

Most of the calibration methods mentioned above require specific calibration targets. This limits their adaptability and introduces the trouble of setting up calibration scenes as well. On the other side, the scene-independent ones either require manual matching or not robust enough. Besides, all of them only focus on two-modal calibration. It's inconvenient to use them sequentially when types of sensors increase.

To solve these problems, we contribute an automatic, targetless, all-in-one calibration pipeline for stereo cameras, thermal cameras, and LiDAR sensors. For the stereo-laser calibration, we directly use Generalized-ICP (GICP) [11] to register their point clouds. To calibrate the thermal camera, we expand Levinson and Thrun's [5] edge alignment algorithm. Our method projects the edge points in both stereo and laser point clouds to thermal images and optimizes the thermal extrinsic to minimize the edge alignment error. Since the edges are widely existed in almost every scenario, our method works in arbitrary environments. Besides, it calibrates the three sensors in one system. This greatly simplifies the cross-modal multi-sensor calibration process.

II. METHODOLOGY

The goal of our algorithm is to take a series of n synchronized stereo image pairs $I_{1:n}^{left}, I_{1:n}^{right}$, thermal images $I_{1:n}^{thermal}$, and laser point clouds $C_{1:n}^{laser}$, captured in arbitrary scenes, and automatically optimize the initial guesses of the

6-DoF rigid-body transformations to get accurate calibrations. The transformation is defined by six parameters $\xi = \{x, y, z, roll, pitch, yaw\}$, where x, y, z are translations, and $roll, pitch, yaw$ are Euler angle rotations. We take the stereo left camera's coordinate system \hat{S} as the base coordinate system, and calibrate other sensors to \hat{S} . As the stereo right to left transformation T_{LR} can be easily obtained with OpenCV [10], there are two transformations remaining to be estimated: laser to stereo transformation T_{SL} and thermal to stereo transformation T_{ST} . We assume that the stereo and thermal cameras' lens distortions have been calibrated so the pinhole camera model is applicable, and their intrinsic matrices K^{left}, K^{right} , and $K^{thermal}$ are obtained.

Our calibration pipeline has three steps. First, generate stereo point clouds by image feature matching and triangulation. Then, calibrate the lasers to stereo by point cloud registration. Finally, optimize the thermal-stereo transformation T_{ST} by minimizing the edge alignment error. The flow of the pipeline is shown in Fig. 1.

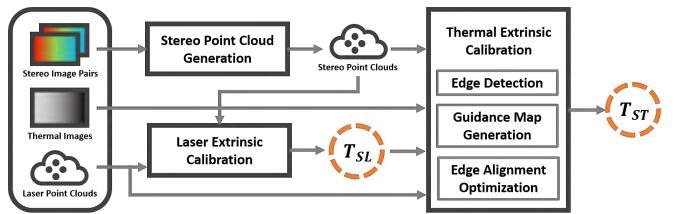


Fig. 1. Flow chart of our calibration pipeline. It takes stereo image pairs, thermal images and laser point clouds as input, and automatically calibrate the stereo-laser and stereo-thermal extrinsic parameters in one run.

A. Stereo point cloud generation

We generate point clouds from stereo image pairs to make full use of their 3D information in later calibrations. The Scale Invariant Feature Transform (SIFT) feature detection algorithm [12] is employed to extract key points and compute descriptors on each stereo image pair $(I_i^{left}, I_i^{right})$, $i = 1 \dots n$. A RegionsMatcher implemented in OpenMVG [13] performs feature matching. Then triangulate matched features to get the 3D positions $P_{i,j}^{stereo}$, where j is the index of matched features in image pair i . We use the OpenCV's [10] triangulation function here. The stereo point clouds $C_i^{stereo} = \{P_{i,j}^{stereo}\}$, $i = 1 \dots n$.

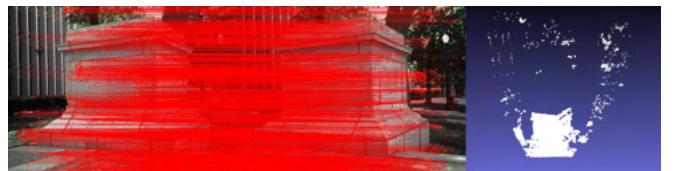


Fig. 2. Example of a matched stereo image pair (left) and the generated stereo point cloud (right).

B. Stereo-laser calibration

The aim of this part is to estimate the laser to stereo transformation T_{SL} . We employ the Generalized-ICP (GICP) [11]

algorithm to register the corresponding stereo and laser point clouds ($C_i^{stereo}, C_i^{laser}$), $i = 1 \dots n$. GICP is a variant of the Iterative Closest Point (ICP) [14] algorithm. It optimizes not only the point-to-point but also point-to-line and point-to-plane distance. The initialization for GICP can be obtained by manually aligning the stereo and laser point clouds in Blender [15] or other 3D processing software. The final T_{SL} is the average of each frame's calibrated transformation.

C. Thermal extrinsic calibration

RGB-thermal automatic calibration is challenging when there are no specially designed calibration targets. Since the RGB and thermal images have different color styles, it's hard to match their features. However, we found that in most cases, the RGB and thermal images share similar edge maps, i.e., if there's an edge on the RGB image, it's very likely to find a corresponding edge on the thermal image. Moreover, as Levinson and Thrun [5] claimed, the correct calibration gives the smallest reprojection edge alignment error (REAE), comparing to the slightly biased calibrations. Based on this, we developed an algorithm to calibrate the thermal extrinsic parameters by minimizing the REAE.

1) Stereo edge points detection: Stereo edge points detection is easy since there are correspondences between the 2D features and 3D points. We first use Sobel operator [16] to detect edges on stereo image pairs (I_i^{left}, I_i^{right}), $i = 1 \dots n$, then pick out the feature points on the edges, and mark their corresponding 3D points as edge points. Fig. 3 gives one example of the stereo edge point detection. The edge points in the i^{th} stereo cloud forms E_i^{stereo} .

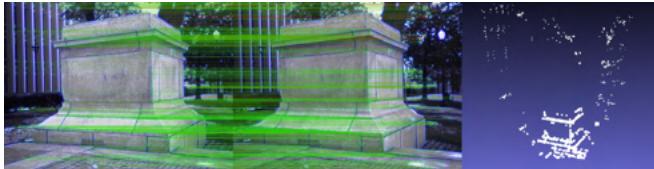


Fig. 3. Example of the matched features on edges in a stereo image pair (left) and the edge points detected in the stereo point cloud (right).

2) Laser edge points detection: Due to the limitation of the number of LiDAR scanning lines, the vertical scanning density is too small for edge detection. Therefore, we only use the horizontal depth difference between the adjacent scans to detect edge points. For a laser point p , we define $left(p)$ as the point generated by p 's left neighbor scan and $right(p)$ as the point of p 's right neighbor scan. The left and right neighbours of p are

$$N_0 = \{left^k(p) \mid k = 1 \dots r\} \quad (1a)$$

$$N_1 = \{right^k(p) \mid k = 1 \dots r\} \quad (1b)$$

where r is the sample radius. The edge is detected according to the distance relationship between p and its neighbors

$$\begin{aligned} p \text{ is edge} \Leftrightarrow & (\forall a \in N_i, dist(a) - dist(p) > \epsilon) \wedge \\ & (\forall b \in N_{1-i}, |dist(p) - dist(b)| \leq \epsilon), i = 0, 1 \end{aligned} \quad (2)$$

where $dist(p)$ is the L2 distance from point p to the origin

$$dist(p) = \sqrt{p.x^2 + p.y^2 + p.z^2} \quad (3)$$

and ϵ is the threshold for edge judgment. Fig. 4 gives one example of the laser edge point detection result. All edge points in the i^{th} laser frame forms E_i^{laser} .

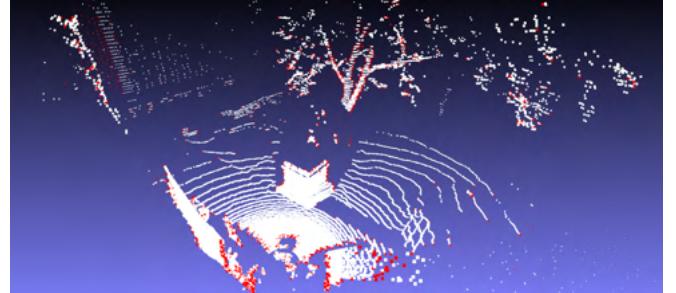


Fig. 4. Example of a frame of laser points. The detected edge points are marked in red.

3) Thermal attraction field map generation: We use the Canny edge detector [17] to detect edges on thermal images $I_i^{thermal}$, $i = 1 \dots n$, then apply distance transform on the edge maps, as shown in Fig. 5. Distance transform calculates the Euclidean distance from each pixel to its nearest edge. We call the distance transformed edge maps "attraction field maps", written as G_i , $i = 1 \dots n$, since their gradient provides the direction to the nearest edge.

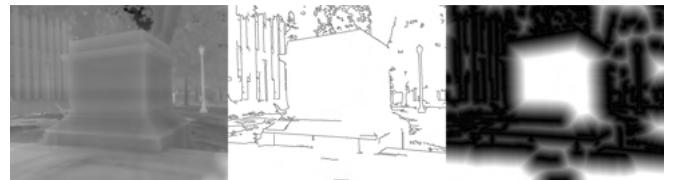


Fig. 5. Example of a thermal image (left), its edge map (middle), and its attraction field map (right).

4) Edge alignment optimization: In this step, we optimize the initial thermal extrinsic to minimize the reprojection edge alignment error (REAE). We first define two projection functions, which project points in stereo local space and laser local space onto thermal images respectively,

$$proj^{stereo}(p) = K^{thermal} \left(T_{ST}^{-1} \begin{bmatrix} p.x \\ p.y \\ p.z \\ 1 \end{bmatrix} \right)_{3D} \quad (4a)$$

$$proj^{laser}(p) = K^{thermal} \left(T_{ST}^{-1} T_{SL} \begin{bmatrix} p.x \\ p.y \\ p.z \\ 1 \end{bmatrix} \right)_{3D} \quad (4b)$$

where T_{SL} , the laser extrinsic, is estimated in laser calibration; T_{ST} , the thermal extrinsic, starts with an initial guess and be optimized iteratively. The notation $(\cdot)_{3D}$ is the conversion from 4-dimensional homogeneous coordinates to

3-dimensional space coordinates.

$$\left(\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \right)_{3D} = \begin{bmatrix} x/w \\ y/w \\ z/w \end{bmatrix} \quad (5)$$

In some cases, the thermal edges are not 100% corresponding to stereo and laser edges. The outliers will mislead the optimizer and negatively affect calibration. Thus, we need to detect and filter them out. We remove the stereo and laser edge points that are too far away from any thermal edge after projecting with the initial transformation, i.e., the inliers are

$$\hat{E}_i^{stereo} = \{p \in E_i^{stereo} \mid G_i[\text{proj}^{stereo}(p)] \leq \delta\} \quad (6a)$$

$$\hat{E}_i^{laser} = \{p \in E_i^{laser} \mid G_i[\text{proj}^{laser}(p)] \leq \delta\} \quad (6b)$$

where δ is the threshold of inliers. Then we can define the cost function, the REAE, as

$$\begin{aligned} \text{REAE} = & \frac{\alpha}{\sum_{i=1}^n |\hat{E}_i^{stereo}|} \sum_{i=1}^n \sum_{p \in \hat{E}_i^{stereo}} G_i[\text{proj}^{stereo}(p)] \\ & + \frac{\beta}{\sum_{i=1}^n |\hat{E}_i^{laser}|} \sum_{i=1}^n \sum_{p \in \hat{E}_i^{laser}} G_i[\text{proj}^{laser}(p)] \end{aligned} \quad (7)$$

where α and β are weight parameters. Considering the quantity of the laser edge points is far less than that of the stereo edge points, we take $\alpha = 1$ and $\beta = 2$.

We use Ceres Solver [18] to minimize the REAE by adjusting the initial guess of T_{ST} , which is obtained by manual measurement with tapes and protractor. Although our cost function is not convex globally, in experiments we found that it is always convex near the correct calibration, thus the algorithm can optimize the extrinsic parameters toward the correct direction with a fine initialization. And as the number of frames n grow larger, the algorithm can tolerate bigger initialization error. That means the user can get the initial guess through rough measurement, and rely on minimizing the REAE to optimize the extrinsic parameters.

III. EXPERIMENTS & RESULTS

A. Sensor setups and data caption

The sensors we used include two Ximea MC124CG-SY RGB cameras, one FLIR Boson® 640 Longwave Infrared (LWIR) thermal camera, and one Ouster OS0-128 LiDAR Sensor. The two RGB cameras are fixed on both sides to form a stereo pair with $baseline = 22.270cm$. The thermal camera is installed between the two RGB cameras, next to the left camera. The LiDAR sensor is mounted a little higher in the middle. The sensors group is mounted under a drone, as shown in Fig. 6.

After installing the sensors, we used tape to measure the translation between the stereo left camera and the thermal camera, got $t_{ST} = [-0.038 \ 0.009 \ 0.001]^T$. As the stereo cameras and thermal camera are fixed on the rod toward the

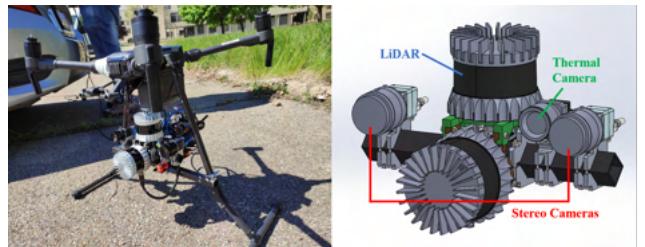


Fig. 6. (left) The complete drone platform. The sensors are mounted under the drone. (right) The detailed setup of the sensors, including stereo cameras, thermal camera, and LiDAR. Notice that there are two LiDAR sensors on our platform, one is horizontal and the other is vertical. We only use the horizontally mounted one.

same direction, we just $R_{ST} = I_{3 \times 3}$. So, the initial guess of T_{ST} is

$$T_{ST}^{initial} = \begin{bmatrix} I_{3 \times 3} & t_{ST} \\ \mathbf{0}_{1 \times 3} & 1 \end{bmatrix}$$

And by manually aligning a pair of stereo and laser clouds in Blender [15], we got the initial guess of T_{SL} .

The data set was captured in the CMU campus, around a stone pier. The recording contains 170 valid frames.

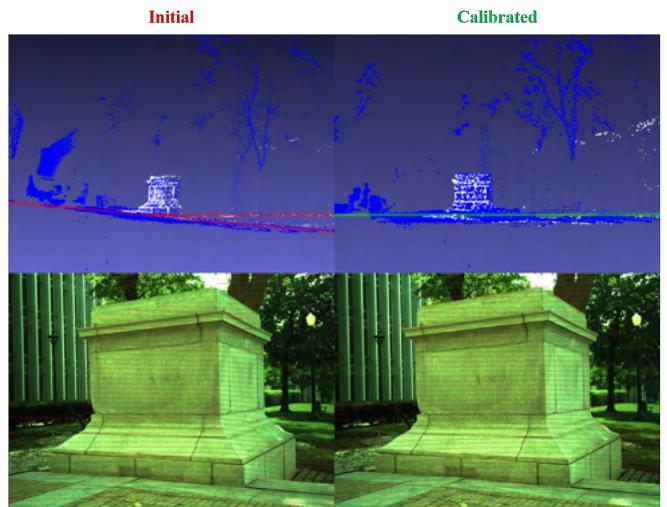


Fig. 7. An example of stereo and laser point cloud registration with initial (left column) and calibrated (right column) laser extrinsic parameters. The first row compares the point clouds alignment in 3D space. When registered with the initial transformation, the ground planes of the two point clouds are at an approximately 7° angle. After the calibration, the two ground planes align well. The second row compares the alignment of the projected laser points and foreground image objects. By viewing the outline of the stone pier in the image, we can see that the initial transformation causes the laser points to deviate a little to the left, while there's no obvious deviation after the calibration.

B. Laser extrinsic calibration result

After registering the laser point clouds with the corresponding stereo point clouds, the two point clouds align well. Both the rotation and translation errors are significantly reduced. We can see this improvement by comparing the registered 3D point clouds with the initial extrinsic parameters and the calibrated ones, as shown in the first row of

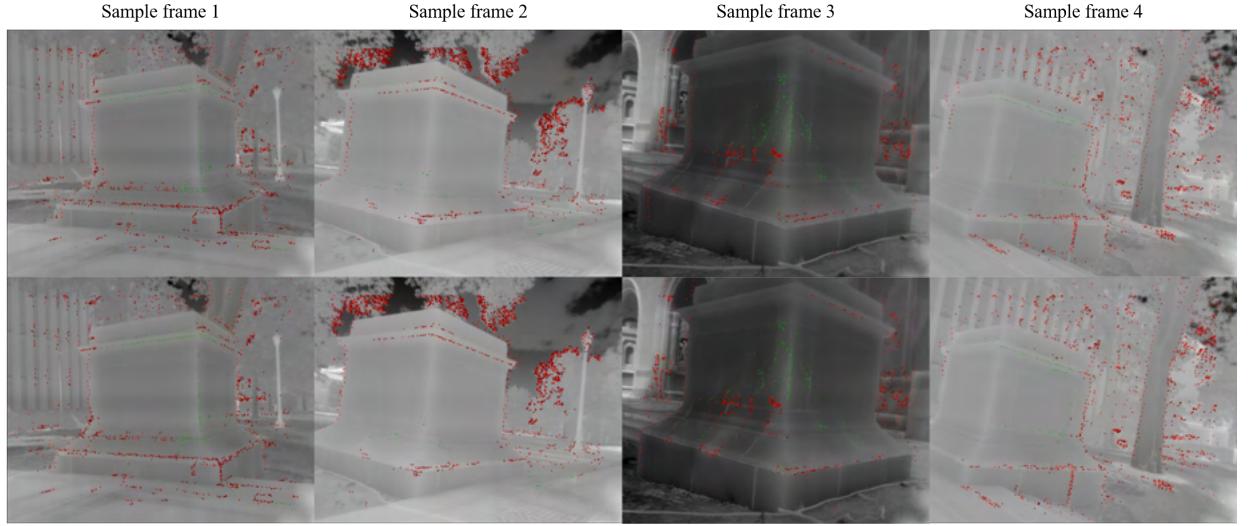


Fig. 8. Four examples of edge point reprojection with initial (first row) and calibrated (second row) thermal extrinsic parameters. In general, the projected edge points fit the thermal edges better after the calibration.

Fig. 7. We can also project the laser points onto the left images, and check the alignment of the projected points and the outline of foreground objects in images. The comparison in the second row of Fig. 7 shows that projecting by the calibrated transformation gives a more accurate reprojection alignment, which indicates an accurate result.

C. Thermal extrinsic calibration result

We picked one frame every ten frames in our "stone pier" outdoor data set, obtained a subset with 17 data frames, and inputted them to the thermal extrinsic optimizer. The REAE drops from 167.136 to 139.647 after the 100 iterations. Fig. 8 shows four edge points reprojection results with initial and calibrated thermal extrinsic parameters respectively. It's clear that the stereo, laser, and thermal edges are aligned better after the calibration.

IV. CONCLUSIONS

This paper presented a multi-modal calibration pipeline for stereo cameras, thermal cameras, and lasers. We first use GICP to calibrate stereo cameras and LiDAR by aligning their point clouds, then project registered stereo and laser point clouds onto thermal images and optimize the thermal extrinsic parameters by minimizing the edge alignment error. Our calibration pipeline can perform automatically in arbitrary environments, and estimate three rigid-body transformations between each pair of the three sensors in one run. It's easy to use since the user only needs to provide a rough initial guess of the transformation and synchronized data frames from the three sensors. The system greatly reduces the complexity of the calibration process, and gives better extrinsic calibration results for high level tasks, such as 3D reconstruction, day-night visual odometry, etc.

Future works will focus on exploiting the real-time performance potential of the system by pre-filtering irrelevant information and improving the parallelism capability. The

real-time calibrator can be used to continuously detect and correct the extrinsic parameters' changes due to the sensor drift or time misalignment. Besides, more kinds of edge information, such as the edge of laser reflectivity map, can be integrated while optimizing cloud-image edge alignment to improve the system's accuracy and robustness.

ACKNOWLEDGMENT

This work is supported by the CMU Robotics Institute Summer Scholars (RISS) program, the CMU AirLab, the Chinese University of Hong Kong, Shenzhen, and the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS). Taimeng would like to thank Rachel Burcin, John Dolan and the RISS team for their support during this research project.

REFERENCES

- [1] Q. Zhang and R. Pless, "Extrinsic calibration of a camera and laser range finder," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, 2004, pp. 2301–2306.
- [2] D. Scaramuzza, A. Harati, and R. Siegwart, "Extrinsic self calibration of a camera and a 3d laser range finder from natural scenes," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 4164–4169.
- [3] P. Núñez, P. Drews Jr, R. P. Rocha, and J. Dias, "Data fusion calibration for a 3d laser range finder and a camera using inertial data," in *European Conference on Mobile Robots (ECMR)*, 2009, pp. 31–36.
- [4] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, "Automatic targetless extrinsic calibration of a 3d lidar and camera by maximizing mutual information," in *AAAI Conference on Artificial Intelligence*, 2012, pp. 2053–2059.
- [5] J. Levinson and S. Thrun, "Automatic online calibration of cameras and lasers," in *Robotics: Science and Systems*, 2013, pp. 29–36.
- [6] C. Guindel, J. Beltrán, D. Martín, and F. García, "Automatic extrinsic calibration for lidar-stereo vehicle sensor setups," in *IEEE international conference on intelligent transportation systems (ITSC)*, 2017, pp. 674–679.
- [7] A. Dhall, K. Chelani, V. Radhakrishnan, and K. M. Krishna, "Lidar-camera calibration using 3d-3d point correspondences," *Computing Research Repository (CoRR)*, 2017. [Online]. Available: <http://arxiv.org/abs/1705.09785>

- [8] Y. Li, J. Tan, Y. Zhang, W. Liang, and H. He, “Spatial calibration for thermal-rgb cameras and inertial sensor system,” in *IEEE International Conference on Pattern Recognition (ICPR)*, 2018, pp. 2295–2300.
- [9] S. S. Shivakumar, N. Rodrigues, A. Zhou, I. D. Miller, V. Kumar, and C. J. Taylor, “Pst900: Rgb-thermal calibration, dataset and segmentation network,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9441–9447.
- [10] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.
- [11] A. Segal, D. Haehnel, and S. Thrun, “Generalized-icp,” in *Robotics: science and systems*, vol. 2, no. 4, 2009, p. 435.
- [12] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [13] P. Moulon, P. Monasse, R. Perrot, and R. Marlet, “Openmvg: Open multiple view geometry,” in *International Workshop on Reproducible Research in Pattern Recognition*. Springer, 2016, pp. 60–74.
- [14] B. PaulJ and M. NeilD, “A method for registration of 3-d shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.
- [15] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: <http://www.blender.org>
- [16] N. Kanopoulos, N. Vasanthavada, and R. L. Baker, “Design of an image edge detection filter using the sobel operator,” *IEEE Journal of solid-state circuits*, vol. 23, no. 2, pp. 358–367, 1988.
- [17] J. Canny, “A computational approach to edge detection,” *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.
- [18] S. Agarwal, K. Mierle, and Others, “Ceres solver,” <http://ceres-solver.org>.
- [19] R. B. Rusu and S. Cousins, “3D is here: Point Cloud Library (PCL),” in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 9-13 2011.

Reason & Act : A Modular Approach to Explanation Driven Agents for Vision and Language Navigation

Shaunak Halbe¹, Ingrid Navarro² and Jean Oh²

Abstract—Vision-and-Language Navigation (VLN) is a multimodal task where an agent follows natural language instructions to navigate in photo-realistic environments. VLN assumes discrete motion along viewpoints of an undirected navigation graph. However, navigation in the real world demands continuous movement through low-level actions, thus motivating the task of Vision-and-Language Navigation in Continuous Environments (VLN-CE). Current approaches to VLN-CE use end-to-end models that attempt to solve both global reasoning and low-level control tasks. Training a single model to perform tasks with vastly differing requirements is difficult. We present the design of a modular system in the form of a global and local planner. The global planner would be responsible for the overall navigation to the desired goal position as indicated by the natural language instruction. It predicts a high-level waypoint to be reached by a local planner through execution of a series of low-level actions. The current baselines for VLN-CE are weak and cannot be scaled for global planning. In this paper, we focus on improving multi-modal understanding of VLN-CE agents with an intention of extending them to form the global planner. To boost multi-modal understanding, we introduce a grounding module along with a Reason-and-Act strategy requiring the agent to identify salient objects in its surroundings. Such a scheme allows the agent to derive visual cues and match them with the verbal indicators given in the instruction. We believe, an agent that can learn to link the signals present in different modalities can perform better in unseen environments.

Index Terms—Vision-and-Language Navigation, Embodied Agents, Hierarchical Planning

I. INTRODUCTION

A robot that can understand and execute human instructions has been a dream for scientists since ages. Up until a few years ago, such a robot was only imagined in science fiction movies. Vision-and-Language Navigation (VLN) [1] takes a significant step towards achieving this dream by formally defining this task. VLN requires an agent to navigate across photo-realistic visual scenes by inferring directional cues from a natural language instruction. Although this is an inherently challenging task for robots to carry out, certain assumptions have simplified the requirements for developing such an agent. VLN agents move by snapping across discrete viewpoints of an undirected navigation graph and are not concerned with the low-level path planning required to reach any viewpoint. These agents observe the environment

¹S. Halbe is with the Department of Computer Engineering, College of Engineering, Pune, Maharashtra, India. halbesa18.coep@coep.ac.in

²I. Navarro and J. Oh are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA., USA ingridn@cs.cmu.edu, jeanoh@nrec.ri.cmu.edu

through panoramic images, and use it to choose the next viewpoint from a list of available candidates.

Some of these assumptions are strong as compared to real world conditions. The more recently proposed task of Vision-and-Language Navigation in Continuous Environments (VLN-CE) [2] takes a step closer to the real world setting, by requiring the agents to execute low-level actions in continuous environments. This setting presents further challenges as the agents are no longer guaranteed perfect localization, actuation, and navigation. The authors of VLN-CE [2] introduce two end-to-end models to serve as baselines. Due to the complex nature of this task, the models achieve low success rates.

We believe that solving such a complex, multi-stage task requires a hierarchical approach with modular components that divide task responsibilities (e.g., alignment, reasoning, control, etc) among themselves. Toward this end, we explore methods for improving the high-level planning aspect. Specifically, we focus on improving the alignment between visual and verbal signals with a goal of leveraging it to improve high-level navigation.

We discuss the structure of a global planner which is entrusted with the task of correlating the visual observations with the instruction and providing us with a high level waypoint to navigate towards. Such a waypoint would then be reached by a local planner through the execution of a series of low-level actions. We explore the idea of an agent that can identify salient features in visual scenes and link them to verbal indicators to develop a richer understanding of the environment. In this spirit, we introduce a reasoning component, which requires the model to identify salient objects in its surroundings that are pivotal in navigating towards the goal. To summarize, our contributions to improve the high-level planning are two fold; we

- introduce a Vision-Language grounding module that generates strongly grounded features in Vision, Depth and Language Space.
- propose a reasoning component that allows an agent to enhance its multi-modal understanding.

II. RELATED WORK

A. Vision-Language Navigation

In VLN [1], an agent is required to follow a navigation instruction from a start location to a goal. Usually, the goal position is not explicitly provided and is to be inferred from the instruction. Overall, VLN models ([3]–[6]) have seen considerable progress in their ability to reach the goal and

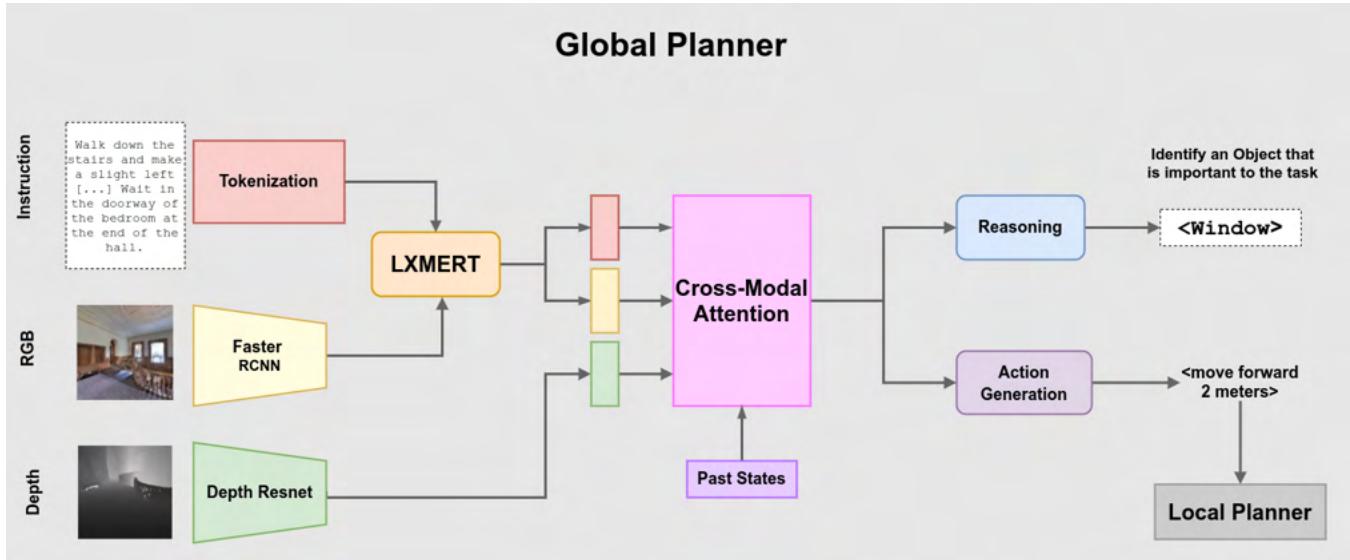


Fig. 1. Global Planner with Cross Modal Attention

the degree of their instruction-trajectory alignment. While most existing works only consider the scenario where the test environments are previously unexplored, some ([3], [5], [7]) also consider a setting where the agent can explore the test scenes prior to evaluation. Among these works, the *Speaker-Follower* [3] approach is quite common where a *speaker* model generates novel instructions from sampled trajectories which are then used for path selection while training the follower. Different from these, transformer based Vision-and-Language pre-training approaches have been successfully extended to VLN. [6], [8] have achieved positive results by demonstrably increasing the vision and language alignment by transferring multimodal transformers pretrained on internet data to VLN settings.

B. Other Language-Guided Navigation Tasks

Apart from VLN, several other tasks involving Language-Guided Navigation / Interaction ([2], [9]–[13]) have been proposed which place an agent in an embodied setting requiring visio-linguistic understanding. These tasks are in a similar vein to the VLN tasks but differ in the activity expected from the agent. Most similar to VLN is VLN-CE [2], which requires an agent to move in a continuous environment in the absence of a navigation graph. VLN-CE also differs with regard to the topological and positional knowledge that the agent has access to. However, VLN-CE has the same high-level objective of language guided navigation as VLN. On the other hand, Embodied Question Answering (EQA) [9] requires an agent to navigate based on the natural language question and answer it using the explored information. Similarly, in Embodied Object Referral (EOR) the agent is tasked with navigating towards an object in the environment based on a natural language instruction. Unlike EOR and EQA, tasks like Vision and Dialog History Navigation (VDHN) and Embodied Goal-Directed Manipulation (EGM) require interaction with the

oracle or manipulation in the environment. We refer readers to [14] for further details about the aforementioned tasks. In this paper we focus on designing a modular agent for the VLN-CE task.

C. Modular Planning

Previous works([12], [15], [16]) have proposed hierarchical approaches to solve Embodied Vision-and-Language Planning tasks. [17] propose *MoViLan*, a modular approach for long horizon tasks such as Vision-and-Language Navigation. *MoViLan* uses a novel Graph Convolutional Neural Network (G-CNN) based approach for mapping by approximating the geometry of nearby objects. The navigation map thus generated is used along with semantic information to predict high-level actions. Finally these high level actions are decomposed into low-level actions using a non-learning search strategy like A*. [15] and [16] use supervised learning to learn to predict high-level waypoints using images and instructions. In the second stage, Reinforcement Learning is used to learn actions to reach these waypoints.

Similar to these approaches, we discuss a modular design to optimize for subgoals using a global planner. However, in VLN-CE environment subgoals are not explicitly provided making it a challenging task to work on.

III. PROBLEM FORMULATION

Following the definition in [14], let $\mathcal{S} = \{\mathcal{V}, \mathcal{L}\}$ represent the set of states encompassing the visual observations, \mathcal{V} , and language inputs, \mathcal{L} . Next, let $\mathcal{A} = \{\text{stop}, \text{turn_left}, \text{turn_right}, \text{move_forward}\}$ include the set of possible actions. The VLN task can be formulated as $\Phi_{\text{VLN}} = \{\mathcal{S}, \mathcal{A}, s_0, s_{\text{goal}}\}$, where $s_0, s_{\text{goal}} \in \mathcal{S}$ represent the initial and target states, respectively. Thus, a plan $\Psi_{\text{VLN}} = \langle s_0, a_0, s_1, a_1, \dots, s_T, a_T \rangle$ exists such that each state s_t , where $t \in [0, T]$, is associated with a location in the environment leading to the final goal. An episode in

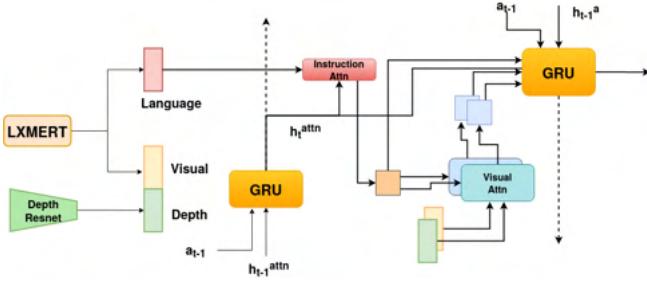


Fig. 2. Cross Modal Attention

VNL requires an agent to find a route from the start state to a target state following an instruction $l \in \mathcal{L}$. At each time-step t , the agent in the environment \mathcal{E} is said to be in a state s_t , represented as (v_t, l) where v_t corresponds to the current visual observation, and l is the instruction. The agent must predict a solution $\hat{\Psi}_{VNL} = \langle s_0, a_0, s_1, a_1, \dots, s_T, a_T \rangle$ by executing an action $a_t \in \mathcal{A}$ at each state s_t following a policy π parametrized by θ such that $a_t = \pi(s_t; \theta)$.

The episode is deemed successful if the sequence of actions, both, delivers the agent close to the intended goal location s_{goal} , and minimizes the difference between the ground-truth plan Ψ_{VNL} , and the predicted plan $\hat{\Psi}_{VNL}$.

IV. APPROACH

In this section, we introduce our global planner π_{global} which is tasked with predicting the next waypoint given visual observations and a specified instruction. We assume the predicted waypoints are passed to a local planner π_{local} which predicts the sequence of low-level actions to reach each of the intended waypoints. As mentioned in the previous sections, our paper focuses on exploring techniques to improve the high-level planning. Thus, we leave local planning out of the scope of our work. We refer readers to Figure 1 for the model architecture.

A. Global Planner

Following [2], we leverage imitation learning [18] to train the global policy π_{global} to predict the next waypoint g_t by imitating expert actions. We train our global agent using AI Habitat [19] and the VLN-CE dataset [2]. In our setting, π_{global} receives an instruction and at each time-step has access to visual observations comprised by color and depth images. The global policy then uses this information to predict the next waypoint $g_t = \pi_{global}(s_t; \theta_{global})$ in terms of distance and heading relative to its current position.

Our global planner is comprised by two sub-modules, a grounding module tasked with ensuring the alignment between the language and visual modalities, and a reasoning component which ensures the agent is able to explain the actions taken in the past. Below we provide further details about the aforementioned modules.

1) *Grounding Module:* In VLN-CE [2], the authors propose a vanilla Sequence-to-Sequence (Seq2Seq) model and a Cross-Modal Attention (CMA) based Recurrent Neural Network (RNN) to serve as baselines for the tasks. Pre-trained

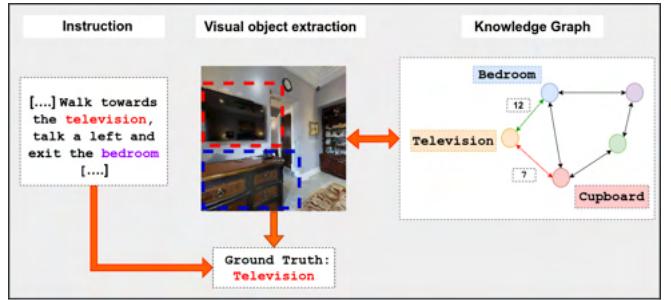


Fig. 3. Extracting the ground truth from Scene Priors

transformer models ([20]–[22]) have starkly outperformed RNN based approaches across a range of language only (Question-Answering, Language Modelling) and multimodal (VQA [23], VisDial [24]) tasks. Drawing inspiration from such tasks, we introduce a Vision-and-Language Grounding module in the form of a pre-trained LXMERT encoder [25]. This module completely replaces the individual instruction encoder and RGB image encoder from the baselines. By design, said model requires bounding box feature vectors of top 36 objects extracted by a 101-layer Faster-RCNN. Thus, we use a pre-trained Faster-RCNN [26] model to extract objects features from our RGB observations to feed to the transformer-based model. We freeze the parameters of both of the encoders, and merely use them as feature extractors. The LXMERT model encodes the image features and instruction tokens and performs cross-modal as well as self-attention over 9 language, 5 visual and 5 cross-modal transformer-encoder layers. For each image-instruction pair, we extract the last layer’s outputs from language and vision streams of LXMERT and combine it to form a representation grounded in vision and language. Separately, we use a Resnet [27] encoder trained on the dataset from the Gibson Environment [28] to extract features from depth observations.

Finally, we use CMA as in [2] to fuse the grounded features extracted from LXMERT with the depth features. CMA consists of two RNN encoders as shown in Figure 2, one to track visual observations and the other one to make decisions based on attended features. The previous action features along with the hidden state are used to attend over the language embedding from LXMERT. This attended language embedding is in-turn used to attend to the visual and depth features. Thus through cross-modal interaction, a strongly grounded representation is produced.

2) *Reasoning Component:* Through this component, we task the agent with identifying an object in its field of sight that is most relevant to instruction and the region that the agent is in. We implement the reasoning module as a linear layer on top of the attention module. We pass the grounded features through the linear layer with an aim to classify it between the 41 object categories present in Matterport3D [29]. We use Cross Entropy as the loss function and optimize it auxiliary to the action prediction loss. Curating good-quality ground truths for each scene is very crucial. At each step, we choose the ground truth object in one of following

	Val-Seen							Val-Unseen						
	PL ↓	NE ↓	nDTW ↑	OS ↑	SR ↑	SPL ↑	ST ↓	PL ↓	NE ↓	nDTW ↑	OS ↑	SR ↑	SPL ↑	ST ↓
Seq2Seq w/o reasoning	7.60	8.48	45.60	30.20	23.52	22.41	102	7.77	9.14	40.73	25.28	16.53	15.03	97
Seq2Seq w/ reasoning (Ours)	8.30	8.66	44.95	34.83	23.65	21.99	99	7.50	8.88	42.40	23.54	16.47	15.28	87

TABLE I
REASONING EXPERIMENT

	Val-Unseen						
	PL ↓	NE ↓	nDTW ↑	OS ↑	SR ↑	SPL ↑	ST ↓
CMA	8.59	9.20	41.49	28.16	17.45	15.82	114
LXMERT + CMA (Ours)	8.31	9.02	42.21	27.19	17.18	15.92	100

TABLE II
ALIGNMENT EXPERIMENT

three ways:

- An object directly mentioned in the instruction is present in the agent’s field of sight
- An object present in the visual scene is correlated to an object mentioned in the instruction
- An object present in the visual scene is often observed in the region (room) where the agent is currently located

We use a Knowledge Graph from Visual Genome[30] to find associations between objects and determine their co-occurrence. We filter this Knowledge Graph by keeping only the objects and regions present in Matterport. Given an object, we use the Knowledge Graph to find other commonly associated objects. While choosing the ground truth an object directly mentioned in the instruction is given highest preference. In case of multiple objects, we use co-occurrence values to determine the ground truth. At each step we maintain a list of objects consisting of the ones directly mentioned in the instructions, associated with the objects mentioned in the instruction and ones that are associated with the region (room) where the agent is present. We extract co-occurrence values between two objects and between an object and a region from the Knowledge Graph. We select the object with the maximum co-occurrence value as the ground truth for the reasoning task.

V. EXPERIMENTS

A. Metrics

We report standard metrics for visual navigation tasks defined in [1], [31], [32] of success rate (SR), success weighted by inverse path length (SPL), normalized dynamic-time warping (nDTW), path length in meters (PL), oracle success rate (OS), navigation error in meters from goal at termination (NE), and steps taken (ST) to quantify the performance of the model.

B. Implementation Details

We train our agents on the ‘train’ split from VLN-CE dataset in the AI Habitat simulator[19]. We utilize the Adam optimizer [33] with a learning rate of 2.5×10^{-4} . We use a DAgger-like [18] approach to collect trajectories with oracle actions as ground truth actions. We collect 10,819 trajectories for both of the experiments. Imitation learning is then performed for 15 epochs over all collected trajectories. In order to match the original setup [2], we set the forward actuation of the agent to 0.25 meters and a turning angle of 15°. We report the results on the entire ‘val-seen’ and ‘val-unseen’ splits from [2].

As mentioned in Section IV-A.1, the grounding module is frozen during the training and inference. We use a Faster-RCNN model pre-trained on Visual Genome [30], to extract 20 object proposals from the RGB image observations. We use the LXMERT model adapted from huggingface [34] pre-trained on multiple multi-modal datasets (MS-COCO [35], VQA [23], GQA [36], and Visual Genome). The depth observations are separately extracted from a Resnet encoder which is updated during training. The textual instructions are tokenized to word-piece embeddings through the LXMERT tokenizer from huggingface, and then pooled out. Finally, we implement our models in PyTorch on top of AI Habitat.

VI. RESULTS

Tables I & II present a comparison of our approach against the baseline models for the Reasoning and the Vision-Language grounding experiments respectively.

A. LXMERT CMA v/s Baseline CMA

We observe that the LXMERT model marginally outperforms the baseline under the metrics of PL, NE, nDTW, SPL and ST. However, its performance drops slightly under the

metrics of OS and SR. Such a moderate performance by LXMERT is counter-intuitive considering the large gains it furnishes on other Vision and Language tasks. The basic LXMERT model contains around 300 million parameters which is far more than the CMA or Seq2Seq baselines. Training LXMERT in an embodied in-simulation setting takes very long adding to the difficulty of achieving or even assessing convergence. This, limited our studies to using a pretrained LXMERT model without fine-tuning its parameters during the VLN-CE training process. We ascribe the middling performance of LXMERT to the domain shift between the high-quality images it was pre-trained on and the significantly lower-quality visuals it experienced through the simulator. A promising future direction would be to replace the Faster RCNN from the LXMERT pipeline with a simpler, more efficient feature extractor and training the overall model on scenes from VLN-CE.

B. Seq2Seq w/ reasoning v/s Seq2Seq w/o reasoning

The agent equipped with the reasoning component achieves comparable results for the val-seen split, which contains scenes observed by the agent during training. The gains with the reasoning component are better realized for the val-unseen split, where it improves over the baseline for majority of the metrics. Although the improvements are minor, they help support our claim of the reasoning component allowing the model to generalize to unseen environments. The reasoning component described in this paper is a preliminary implementation of our idea. We plan to pursue more sophisticated mechanisms for inducing reasoning in the agent.

VII. CONCLUSIONS

In this work, we proposed the idea of a modular agent for VLN-CE. However, we focused on the high-level planning component of this agent. More specifically, we worked towards improving the baselines and presented a strategy to incorporate them into a modular architecture. Although the results are mixed and the gains are smaller, these directions appear to be promising and create ample opportunities for development in the future. Following this work, we would like to improve the Vision & Language Grounding module, by making it computationally efficient, thus allowing it to be trained or finetuned on VLN-CE episodes.

Further we would like to explore better alternatives for inculcating the ability of reasoning in such agents by allowing them to explore and understand the environments. Finally, we plan to build and test the entire modular agent by integrating the proposed high-level policy with a local planner.

ACKNOWLEDGMENT

Shaunak Halbe thanks the Bot Intelligence Group for their immense support, particularly Ingrid Navarro and Dr. Jean Oh for their mentorship and guidance. He would like to thank Felix Labelle for the invaluable inputs he offered during the project development. He is also grateful to RISS program advisors Rachel Burcin and Dr. John Dolan for

their consistent support and for organizing the RISS 2021 program. Finally he would like to express his gratitude towards the sponsors for supporting him during the summer.

REFERENCES

- [1] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel, “Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments,” 2018.
- [2] J. Krantz, E. Wijmans, A. Majumdar, D. Batra, and S. Lee, “Beyond the nav-graph: Vision-and-language navigation in continuous environments,” 2020.
- [3] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell, “Speaker-follower models for vision-and-language navigation,” 2018.
- [4] V. Jain, G. Magalhaes, A. Ku, A. Vaswani, E. Ie, and J. Baldridge, “Stay on the path: Instruction fidelity in vision-and-language navigation,” 2019.
- [5] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Y. Wang, and L. Zhang, “Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation,” 2019.
- [6] A. Majumdar, A. Shrivastava, S. Lee, P. Anderson, D. Parikh, and D. Batra, “Improving vision-and-language navigation with image-text pairs from the web,” 2020.
- [7] H. Tan, L. Yu, and M. Bansal, “Learning to navigate unseen environments: Back translation with environmental dropout,” 2019.
- [8] Y. Hong, Q. Wu, Y. Qi, C. Rodriguez-Opazo, and S. Gould, “A recurrent vision-and-language bert for navigation,” 2021.
- [9] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, “Embodied question answering,” 2017.
- [10] D. Gordon, A. Kembhavi, M. Rastegari, J. Redmon, D. Fox, and A. Farhadi, “Iqa: Visual question answering in interactive environments,” 2018.
- [11] J. Thomason, M. Murray, M. Cakmak, and L. Zettlemoyer, “Vision-and-dialog navigation,” 2019.
- [12] M. Shridhar, J. Thomason, D. Gordon, Y. Bisk, W. Han, R. Mottaghi, L. Zettlemoyer, and D. Fox, “Alfred: A benchmark for interpreting grounded instructions for everyday tasks,” 2020.
- [13] Y. Qi, Q. Wu, P. Anderson, X. Wang, W. Y. Wang, C. Shen, and A. van den Hengel, “Reverie: Remote embodied visual referring expression in real indoor environments,” 2020.
- [14] J. Francis, N. Kitamura, F. Labelle, X. Lu, I. Navarro, and J. Oh, “Core challenges in embodied vision-language planning,” *CoRR*, vol. abs/2106.13948, 2021. [Online]. Available: <https://arxiv.org/abs/2106.13948>
- [15] D. Misra, A. Bennett, V. Blukis, E. Niklasson, M. Shatkin, and Y. Artzi, “Mapping instructions to actions in 3d environments with visual goal prediction,” 2019.
- [16] V. Blukis, D. Misra, R. A. Knepper, and Y. Artzi, “Mapping navigation instructions to continuous control actions with position-visitation prediction,” 2018.
- [17] H. Saha, F. Fotouhif, Q. Liu, and S. Sarkar, “A modular vision language navigation and manipulation framework for long horizon compositional tasks in indoor environment,” 2021.
- [18] S. Ross, G. J. Gordon, and J. A. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” 2011.
- [19] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, “Habitat: A platform for embodied ai research,” 2019.
- [20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2017.
- [21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” 2019.
- [22] J. Lu, D. Batra, D. Parikh, and S. Lee, “Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks,” 2019.
- [23] A. Agrawal, J. Lu, S. Antol, M. Mitchell, C. L. Zitnick, D. Batra, and D. Parikh, “Vqa: Visual question answering,” 2016.
- [24] A. Das, S. Kottur, K. Gupta, A. Singh, D. Yadav, J. M. Moura, D. Parikh, and D. Batra, “Visual Dialog,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

- [25] H. Tan and M. Bansal, “Lxmert: Learning cross-modality encoder representations from transformers,” 2019.
- [26] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” 2016.
- [27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *arXiv preprint arXiv:1512.03385*, 2015.
- [28] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese, “Gibson Env: real-world perception for embodied agents,” in *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018.
- [29] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, “Matterport3d: Learning from rgb-d data in indoor environments,” 2017.
- [30] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. S. Bernstein, and F.-F. Li, “Visual genome: Connecting language and vision using crowdsourced dense image annotations,” 2016.
- [31] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, and A. R. Zamir, “On evaluation of embodied navigation agents,” 2018.
- [32] G. Ilharco, V. Jain, A. Ku, E. Ie, and J. Baldridge, “General evaluation for instruction conditioned navigation using dynamic time warping,” 2019.
- [33] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2017.
- [34] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, “Huggingface’s transformers: State-of-the-art natural language processing,” 2020.
- [35] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, “Microsoft coco: Common objects in context,” 2015.
- [36] D. A. Hudson and C. D. Manning, “Gqa: A new dataset for real-world visual reasoning and compositional question answering,” 2019.
- [37] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: <http://arxiv.org/abs/1707.06347>
- [38] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, “Matterport3d: Learning from rgb-d data in indoor environments,” *International Conference on 3D Vision (3DV)*, 2017.
- [39] E. Wijmans, A. Kadian, A. Morcos, S. Lee, I. Essa, D. Parikh, M. Savva, and D. Batra, “DD-PPO: learning near-perfect pointgoal navigators from 2.5 billion frames,” in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [Online]. Available: <https://openreview.net/forum?id=H1gX8C4YPr>

Accurate Pedestrian Localization for Urban Crosswalks

Rayna Hata¹, Issac Isukapati², Stephen Smith²

Abstract—Safe and easy crossing of signalized pedestrian walkways in urban settings still poses a challenge for vision-impaired or mobility-impaired people. To tackle this problem, PedPal, a mobile application, was developed to help those with the app influence the traffic control systems to adjust the length of time allotted for them to cross the walkway. Through the app, the pedestrian may adjust the length of time allotted to them to cross the intersection to fit to their speed. If the intersection is not safe for the pedestrians to cross yet, the app will also notify when it is safe to cross.

Using this app, the pedestrian may adjust the length of time allotted to cross the intersection to fit their speed and signal to them when it is safe to travel across. Because the localization abilities of the iPhone are not accurate enough to conform to the needs of PedPal, there are currently Bluetooth beacons installed into each corner of the intersection to detect which corner the pedestrian is at by having the pedestrian manually interact with the beacon. But, this proves to be a problem if the pedestrian's hands are occupied. Furthermore, the current Bluetooth beacons installed cannot accurately track the pedestrian's progress through the interaction and the exact moment they arrive at the crosswalk. In this paper, we explore different types of radio beacons and algorithms to achieve the best localization accuracy of the pedestrian's location with respect to the corners of the intersection.

I. INTRODUCTION

Mundane methods of transportation, such as walking across an intersection, are a seemingly easy task for most people. However, pedestrians with visual or mobility disabilities find this one of the most daunting tasks of their everyday lives. A survey sent to 1,123 members of the AER Division 9 found that ninety-eight percent of respondents expressed that they found knowing when to start crossing difficult. Ninety-seven percent of their respondents stated that they had difficulty keeping straight on a crosswalk. Furthermore, sixty-six of their respondents said that they have trouble figuring out where the destination corner was. [1]

In an urban setting with other pedestrians, more significant levels of noise, and unpredictable factors, vision-impaired pedestrians tend to spend more time at an intersection to understand the patterns at that particular intersection. Using cues around them such as the noise of an idling car to understand the width, direction, and surroundings before crossing, these pedestrians can spend up to multiple crossing cycles before feeling safe enough to make the cross. Aside from using vehicles around them as cues to help them map out the crosswalk better, the pedestrian may look for the

¹Rayna Hata is with the department of Computer Science, Colby College, Waterville, Maine rhatata23@colby.edu

²Isaac Isukapati and Stephen Smith are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA (isaack.sfs@cs.cmu.edu

curb ramp to find the start of the crosswalk. However, the pedestrians and orientation and mobility specialists find that looking for the curb ramp can cause more harm than good at times. While looking for the ramp, the pedestrian may get misaligned with the intersection and thus veer off in a different direction when crossing. [2]

PedPal is a guided navigation mobile application meant to serve as an assistive technology for successfully navigating crosswalks in an urban setting for blind or mobility disabled pedestrians. PedPal currently utilizes two systems to help accommodate each pedestrian with a more personalized crossing time to ensure that they successfully cross the intersection. The first system is a real-time adaptive signal control using the SURTRAC traffic signal system. SURTRAC, scalable urban traffic control, allows each intersection to decide its green time independently of one another, relying on real-time information of incoming vehicles using videos or radars. A signal timing plan is created using the information and communicated with the nearby intersection signals. This type of traffic signalization allows for efficiency within an urban environment leading to less lost time. [3]

The same idea of individualizing the length of the green light can also be implemented for a pedestrian signal. In order to do so, there must be some form of Pedestrian-to-Infrastructure (P2I) communication. The current form of P2I in the PedPal app comes from the user manually indicating which intersection they intend to cross using either audio or visual options. Upon selecting the intersection, the app will report the state of the intersection to the pedestrian. If the signal is on a green light, the app will read out a countdown of the number of seconds left. Depending on the type of disability that the pedestrian has, the app will request additional time to be added to the length of the green signal to allow the pedestrian to cross with ample time. [4]

Currently, PedPal uses Bluetooth beacons to assess if the pedestrian is approaching the end corner of the intersection. While this is a way to connect the signal and the mobile device, Bluetooth's localization ability to sense the distance between the signal and pedestrian is not accurate enough to fit the needs of PedPal. Without accurate pedestrian localization, the app will not be able to guide the pedestrian successfully across an intersection.

As an alternative to Bluetooth, we will incorporate Ultra-Wideband (UWB) beacons to establish communication between the mobile device and the signal as a method for the localization of pedestrians. By extending the new Nearby Interaction (NI) framework provided by Apple and supplementing the distance measurements with additional data from the Core Motion (CM) framework, the mobile device will

determine the progress and the direction of the pedestrian approaching the end of the intersection.

II. RELATED WORKS

A. Bluetooth for Pedestrian Assistance and Tracking

Blind pedestrian assistance technology is an emerging field with different technological approaches. One method utilizes Bluetooth beacons to communicate with the pedestrian to convey the state of the intersection. Universal Real-Time Navigational Assistance (URNA) connects the user's phone to the Bluetooth beacon placed on the pedestrian light signal to relay the information displayed on the light to the phone screen. The phone, using Text-to-Speech, is designed to read out the state of the light and count down the number of seconds remaining in that cycle. This method effectively ensures that the pedestrian is aware of the state of the traffic signal and the number of seconds left before the green light state is over. [5] One of the difficulties that a vision-impaired pedestrian faces using this interface is that the URNA does not know the exact location of the pedestrian, and thus cannot guide them across it. The system also does not implement a method for collecting information on whether the pedestrian has safely crossed the intersection.

Localization tracking of objects or people has become popular to increase productivity, efficiency, or safety within set environments. One method of a Bluetooth localization system is in an indoor setting to track objects with mobile devices. The mobile devices communicate with signals emitting Bluetooth nodes as the individual moves around the space. A study tested the accuracy of the Bluetooth indoor localization gathered via the communication between the Bluetooth nodes and the mobile device. The study had two stationary time measurement systems with a receiver and a moving transmitter to localize the Bluetooth device to run the test. The final accuracy was found to be around ± 1 meters. [6] While ± 1 meters in an indoor setting can be harmless, this kind of distance on a crosswalk in an urban setting can cause significant safety issues.

B. GPS Pedestrian Assistance

Another proposed method of pedestrian assistance technology for blind pedestrians is using a mobile phone's GPS to track if the pedestrian is near a crosswalk. Using the GPS to assess the pedestrian's coordinates, a satellite image of the location is pulled into a crosswalk detection framework to guide the user toward the intersection. While this method is helpful for real-time positioning of the pedestrian's location, the paper mentions that due to factors such as image acquisition problems or GPS accuracy, the localization accuracy cannot be narrowed down to cm. [7] Furthermore, there is no communication between the mobile device and the signal. Therefore, while the satellite can guide the pedestrian to the desired crosswalk, it cannot notify the pedestrian of the state of the crosswalk, nor can it guide them during the crossing.

C. Use of UWB in localization

UWB localization can be used in various settings. For example, UWB is used to track unmanned aerial vehicles [8], objects in a factory environment to increase automation and efficiency [9], or in indoor pedestrian tracking systems. An indoor pedestrian localization system was created by fusing UWB, pedestrian dead reckoning (PDR), and floor map data. Using time-of-arrival (TOA) from the UWB, step number, step length, direction from PDR, and a floor map, they achieved a localization accuracy of 0.15m. While there are many indoor pedestrian localization methods proposed, there are much fewer outdoor pedestrian methods. The main difference between an outdoor and an indoor setting is the amount of noise change in an outdoor setting. In an indoor setting, obstacles such as walls are stationary. However, in an outdoor setting, there are constantly moving objects with varying degrees of obstruction.

III. METHODOLOGY

There are many other methods of localization tracking using different short-range wireless communication devices (Bluetooth, GPS, WiFi). However, in this paper, we will be discussing the use of UWB devices as a method of pedestrian tracking. UWB beacons are known to have a high accuracy of up to 20 cm in an indoor setting. Another advantage of UWB devices over other short-range communication devices is that the short-duration pulses allow for easier sorting of correct signals vs. reflective signals. [10] Other advantages of UWB over other devices include potentially lower cost and more resistance to severe multi-path and jamming. [11] In an urban setting, where there is potential for a significant level of noise and obstacles, a device with more resistance to signal interruption is needed in order to ensure the most accurate calculation of distance between the anchor node and the mobile node.

While indoor localization of human tracking is more widely researched than outdoor tracking, the implementations are similar. Applications such as PedPal will utilize implementations of UWB devices in urban outdoor settings to assist blind pedestrians in crossing an intersection.

A. Nearby Interaction

Recently, Apple has incorporated UWB chips into their devices as well. These UWB chips, named U1 chips, allow for Apple devices to communicate with other devices that also have U1 chips. Following the release of the U1 chips, Apple has also released an object tracking device called Air-Tags. These small tracking chips communicate with an iPhone and lets the owner track objects attached to the Air-Tags with high precision. This kind of tracking can be implemented into pedestrian tracking as well. With the mobile node being the pedestrian with the iPhone and the anchor being the corner of the pedestrian's intersection, Apple's Air-Tag technology could accurately track the progress. However, an Air-Tag can only be registered to one iCloud account at a time. Therefore, using the AirTag as a solution to the tracking device is not a feasible option.

A solution to this problem would be to implement Apple's Nearby Interaction (NI) feature with an Apple-approved UWB device. Apple has released multiple third-party UWB chip devices compatible with the U1 chips installed in mobile phones. Using the NI framework, the UWB device will be able to get the precise distance of the pedestrian and the direction that the pedestrian is walking. This framework allows for multiple phones to be connected to one device, allowing for multiple pedestrians to cross the intersection while using the UWB device to determine their distance. [12]

B. Core Motion

While the Nearby Interaction framework allows the iPhone to get the distance and the direction from the anchor node, its optimal performance is when the iPhone is held vertically straight and in front of the person holding the phone. Without the proper position, there is a greater likelihood of the distance and direction being incorrect. To combat that, we plan to investigate if supplementing the Nearby Interaction data with Inertial Measurement Unit (IMU) sensors will decrease the potential errors created by the iPhone held incorrectly. Apple's iPhones have IMU sensors built into them, which the Core Motion framework can access. The Core Motion framework allows for apps to access motion-related data generated via the phone's hardware. The iPhone currently has a gyroscope, pedometer, magnetometer, and altimeter that can be utilized. [13]

C. Logging Distance Between an AirTag and iPhone

In order to test the accuracy and the reliability of the U1 chip, we first started with manually collecting the distance measurements displayed on an iPhone when it is connected to a single AirTag. With one person holding the AirTag and another person walking along with a measured distance, we collected distance measurements at different stop points. In the first round, the measured starting distance between the AirTag and the iPhone was 24 ft. While walking towards the AirTag, we stopped every three feet to check for a difference between the actual distance and the distance displayed on the iPhone. To see if the phone's orientation affects the distance measurements, we tested three different ways a phone can be held. (1)Phone held vertically straight, screen parallel to user's chest (2)Phone held horizontally, screen perpendicular to chest (3) Phone held tilted down. These different positions tested out how the distance measurements changed, and direction was affected.

There are often many moving parts in an outdoor setting, which can act as an obstacle between the anchor and the iPhone. To see what obstacles may affect the line of sight readings, we placed a human in between the AirTag and the iPhone. Secondly, we placed a vehicle between the AirTag and the iPhone.

The third experiment that we ran to test the accuracy of the AirTag tested if the distance readings would fluctuate if the pedestrian is stationary. A pedestrian often needs to wait at the signal before crossing, and if the readings bounce

around during that time, then a better localization method is needed. Going out to a set measured distance, the holder of the iPhone stood in one spot to see how the readings would differ over a period of time.

Finally, we tested the maximum distance that an iPhone can detect an AirTag to see what the range was in order to determine if this is a viable option for a large intersection.

IV. RESULTS

A. Testing the Accuracy of Measurement Readings

In this experiment, we found that the accuracy of the AirTag matches the measured distance consistently no matter what the phone's orientation is. Furthermore, the measurements show that the error resolution is closer to cm than ft. The margin of error was found due to the fact that if the phone is exactly 21 ft. from the AirTag, then the iPhone would display the distance as 21 ft. As we moved forward slightly, the distance displayed became 20 ft, rounding down. As the AirTag and the iPhone became approximately half a foot away from each other, the phone would display that we have arrived at the location of the AirTag.

B. Testing with Obstacles

Testing for the accuracy and strength of the readings, we found that placing a human between the AirTag and the iPhone as an obstacle did not affect the accuracy of the measurement. However, the strength was displayed as a weaker signal. When a vehicle was placed between the AirTag and the iPhone, the connection was lost, and measurements were not displayed.

C. Testing the Stability of Readings

Checking for the stability of the distance measurements, we found that the iPhone quickly converged to a single distance after the holder stood still for a few seconds. While standing at the same point, the displayed measurement did not change and stayed accurate for an extended period.

D. Testing the Range

Walking away from the AirTag until the iPhone lost connection, the furthest measured distance was approximately 90 feet. Different situations may affect the range, but, in this instance, the range is large enough to be viable for PedPal.

V. CONCLUSION

Our initial tests indicate that using the Apple-approved UWB device is a reliable form of tracking the pedestrian's distance from the corner of the intersection. The distance measurements measured by the AirTag and the iPhone were accurate and consistent with less than a foot error. This fits the requirements that we have determined as a safe error in order to avoid having the pedestrian miss the intersection.

One problem that would need to be addressed is a scenario when there is an obstacle between the UWB device and the pedestrian. It seems as though there is a problem with getting non-line-of-sight data. While the LOS data is very accurate, the chances of having a good LOS reading at all times like

low as there are many moving parts in an urban setting. This issue can be potentially fixed by fusing the measured distance with the Core Motion data.

A. Current Work

Ongoing research includes progressing to storing a continuous stream of distance measurements between two devices. A continuous data collection allows us to see if there are fluctuations in the data that the iPhone did not display. Furthermore, while storing the distance measurements, we are also storing the Core Motion measurements to find a way to fuse the two data readings from the sensors. Using the data from the Core Motion, we are looking to get better directional measurements as the orientation of the iPhone affects the directional readings to determine if the pedestrian has veered off the intersection. Next, we plan to move to use more than one UWB device at a time to form a method of triangulation instead of relying on one device at all times. Lastly, we plan to progress to our own UWB devices using an Apple-approved third-party device. These devices allow for multiple connection sessions at a time, addressing the problem of more than one person using the UWB device.

ACKNOWLEDGMENTS

This work would not have been possible without the support of Traffic21 and the Robotics Institute Summer Scholars Program. A large thanks to Dr. Stephen Smith and Dr. Isaac Isukapati from the Carnegie Mellon's Intelligent Coordination and Logistics lab for their mentor-ship and support throughout the process. I would also like to thank Rachel Burcin and Dr. John Dolan for their guidance throughout the summer.

REFERENCES

- [1] B. L. Bentzen, J. M. Barlow, and L. Franck, "Addressing barriers to blind pedestrians at signalized intersections," *Institute of Transportation Engineers. ITE Journal*, vol. 70, no. 9, p. 32, 2000.
- [2] B. L. Bentzen, J. M. Barlow, and T. Bond, "Challenges of unfamiliar signalized intersections for pedestrians who are blind: Research on safety," *Transportation research record*, vol. 1878, no. 1, pp. 51–57, 2004.
- [3] S. F. Smith, G. J. Barlow, X.-F. Xie, and Z. B. Rubinstein, "Smart urban signal networks: Initial application of the surtrac adaptive traffic signal control system," in *Twenty-third international conference on automated planning and scheduling*, 2013.
- [4] S. F. Smith, Z. B. Rubinstein, S. Braham, M. B. Dias, and J. Marousek, "Safe intersection crossing through pedestrian-to-infrastructure communication," Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-19-83, May 2019.
- [5] S. Bohonos, A. Lee, A. Malik, C. Thai, and R. Manduchi, "Universal real-time navigational assistance (urna) an urban bluetooth beacon for the blind," in *Proceedings of the 1st ACM SIGMOBILE international workshop on Systems and networking support for healthcare and assisted living environments*, 2007, pp. 83–88.
- [6] G. Fischer, B. Dietrich, and F. Winkler, "Bluetooth indoor localization system," in *Proceedings of the 1st Workshop on Positioning, Navigation and Communication*, 2004, pp. 147–156.
- [7] M. C. Ghilardi, J. C. Jacques, and I. Manssour, "Crosswalk localization from low resolution satellite images to assist visually impaired people," *IEEE computer graphics and applications*, vol. 38, no. 1, pp. 30–46, 2016.
- [8] F. Lazzari, A. Buffi, P. Nepa, and S. Lazzari, "Numerical investigation of an uwb localization technique for unmanned aerial vehicles in outdoor scenarios," *IEEE Sensors Journal*, vol. 17, no. 9, pp. 2896–2903, 2017.
- [9] L. Barbieri, M. Brambilla, A. Trabattoni, S. Mervic, and M. Nicoli, "Uwb localization in a smart factory: Augmentation methods and experimental assessment," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–18, 2021.
- [10] G. Shi and Y. Ming, "Survey of indoor positioning systems based on ultra-wideband (uwb) technology," in *Wireless communications, networking and applications*. Springer, 2016, pp. 1269–1278.
- [11] M.-G. Di Benedetto, "Uwb communication systems: a comprehensive overview," pp. 1–3, 2006.
- [12] "Nearby interaction." [Online]. Available: <https://developer.apple.com/documentation/nearbyinteraction>
- [13] "Core motion." [Online]. Available: <https://developer.apple.com/documentation/coremotion>

GPU Enhanced Front-end for Visual-inertial Odometry

Yao He¹, Huai Yu² and Sebastian Scherer²

Abstract—A robust and versatile visual-inertial odometry (VIO) serves as a fundamental tool for state estimation in a wide range of applications, such as robotic navigation, autonomous driving, and virtual reality. Despite the urgent demands in these applications, the high cost on CPU resource and computation latency limits VIO’s possibility in integration with other applications. Recently, many existing computer vision works have utilized the powerful embedded graphics processing units (GPUs) to improve the information processing capability and reduce the latency. Inspired by these works, we incorporate the GPU-improved algorithms in the field of VIO and thus propose a new front-end for VINS-mono. Typically, we utilize the feature detection and tracking algorithms in the Vision Programming Interface (VPI) provided by NVIDIA. We also test the VIO modules provided in the CUDA Visual Library (VILIB), and implement our own random-sample-consensus (RANSAC) algorithm on GPU. This work shows that without losing the high accuracy of state estimation, the CPU resource occupation rate and computational latency are reduced by 40.4% and 50.6%, respectively.

Index Terms—Monocular visual-inertial systems (VINSS), Visual-Inertial SLAM, state estimation.

I. INTRODUCTION

State estimation is the most fundamental module for a wide range of applications, such as robotic navigation, autonomous driving, and augmented reality (AR) [1]. The state-of-the-art approaches are visual-inertial odometry (VIO) algorithms, which combines inertial measurement units (IMUs) and the cameras to solve the odometry problems accurately. Numerous work has been proposed for this combination, for example, VINS-Mono [1], ROVIO [2], and ORB-SLAM [3] for mono-VIO, ORB-SLAM2 stereo [4] for stereo-VIO. Generally, a VIO algorithm composes of a front-end that processes image data, and a back-end that carries out the optimization of poses. Since the data size processed by the front-end is remarkably large, the cost on CPU usage and computation latency is high. Jeon [5] tests the CPU usage of different VIO algorithms on various NVIDIA hardware (Jetson TX2, Xavier NX, and AGX Xavier boards). His work shows that most of the VIO algorithms have extensive CPU usages on different NVIDIA hardware. The computation latency is also a significant problem for real-time visual processing. It might result in an accumulating delay when the VIO operates for a long time. Approaches are applied to reduce the cost on CPU and computation latency. For example, VINS-mono limits the number of tracked features

¹ Yao He is with the Department of Electronic and Information Engineering, The Chinese University of Hongkong, Shenzhen, China. yaohe@link.cuhk.edu.cn

²Huai Yu, and Sebastian Scherer is with the AirLab, Carnegie Mellon University, Pittsburgh, PA 15213 huaiyu,basti@andrew.cmu.edu

to a certain amount. These approaches work to some extend, but they reduce the accuracy of the estimation since they limit the performance of the front-end. In our work, we would like to build a new front-end for VIO. This front-end reduces the cost on CPU usage and computation latency, while the estimation accuracy maintains or even gets improved at the same time.

We mainly focus on incorporating the graphics processing units (GPUs) enhanced modules into the VIO algorithms. Recently, the powerful embedded GPUs have been utilized to improved the information processing capability and reduce the latency in the field of VIO. Noticeable approaches are Vision Programming Interface (VPI) proposed by NVIDIA [6], and CUDA Visual Library (VILIB) proposed by Nagy, et al. [7]. In our work, we incorporating them into the front-end of VIO and test the performance. We also proposed our own GPU enhanced random-sample-consensus (RANSAC) algorithm [8]. With the help of GPUs, we expect that the CPU usage and computation latency could be reduced, while the estimation accuracy and robustness maintain.

We perform an extensive evaluation on public datasets to compared the accuracy of different versions of GPU enhanced VIOs. The computational latency of the front-end is also measured. In addition, we use our own VIO setup to collect data in large environment, and evaluate the performance of VIO on Xavier AGX.

The rest of this paper is organized as following. Section II provides the relevant literature. Section III describes the GPU algorithms we use and the implementation of RANSAC algorithm on GPU. Section IV provides the testing results with different GPU enhanced algorithms. Finally, this paper is concluded with the discussion and possible future research directions.

II. RELATED WORK

A. VIO Algorithms

Numerous VIO algorithms have been proposed in recent years. Qin [1] et al. proposed VINS-mono. It starts with a robust procedure for estimator initialization. By fusing preintegrated IMU measurements and feature observations, a tightly coupled, nonlinear optimization-based method is performed to obtain accurate state estimation. Combined with the tightly coupled formulation, a loop detection module enables relocalization with minimum computation. Additionally, it performs 4-DOF pose graph optimization to enforce the global consistency. Furthermore, VINS-mono reuses a map by saving and loading it in an efficient way. The current and previous maps are merged together by the global pose graph optimization. In Jeon’s work [5], the CPU usage of

VINS-mono on Xavier is on TX2, Xavier NX, and Xavier AGX are around 150-180%, 230-250%, and 150-170%, respectively. Here the values are above 100% because of multi-core processing. The front-end feature detector consumes 15ms, and KLT tracker consumes 5ms [1].

Bloesch et al. proposed ROVIO [2]. It combines an iterated extended Kalman filter (EKF), a fully robocentric formulation of visual-inertial odometry, and a photometric error model. The landmark tracking by iterated extended Kalman filter allows per-landmark iterative updates and thus provides simultaneous landmark tracking and full state refinement. Bloesch et al. claims that ROVIO can run in real time on computationally constrained platforms. In Jeon's work [5], ROVIO has the least CPU usages (around 60%) among all the tested VIOs. However, EKF is less efficient compared to the bundle adjustment method adopted in other VIOs [9], the computational latency on back-end will be larger in ROVIO.

R. Mar-Artal et al. [3] proposed ORB-SLAM. The ORB-SLAM performs an automatic and robust initialization procedure based on model selection. It utilizes the ORB features which provides good invariance to changes in viewpoint and illumination. The tracking and mapping is focused in a local covisible area, it achieves real time operation in large environments. A survival of the fittest approach to map point and keyframe selection provides robust tracking and enhanced lifelong operation. Built from a spanning tree, the loop closure links strong edges from the covisibility graph to achieve real time optimization. The camera relocalization is also invariant to viewpoint and illumination, which allows recovery from tracking failure and enhances map reuse. Based ORB-SLAM, R. Mar-Artal et al. proposed ORB-SLAM2 [4], the first open-source SLAM system for monocular, stereo, and RGB-D cameras, including loop closing, relocalization, and map reuse. In Jeon's work [5], the CPU usage of ORB-SLAM2 on TX2, Xavier NX, and Xavier AGX are around 150-250%, 200-260%, and 150-240%, respectively.

B. GPU Enhanced VIO Algorithms Libraries

After the proposition of CUDA [10], a parallel computing platform and programming model that leverages the parallel compute engine in NVIDIA GPUs, programmers can solve complex computational problems in a parallel ways. Recently, several works have utilized GPUs to improve the information processing capability and reduce the latency.

NVIDIA proposed the Vision Programming Interface (VPI) [6]. VPI is a software library that implements computer vision and image processing algorithms on several computing hardware platforms, such as CUDA, available in NVIDIA embedded and discrete devices. It supports easy inter-operation with existing projects that make use of OpenCV and NVIDIA® CUDA® SDK libraries. This allows for gradual replacement of existing computing tasks with faster VPI equivalents.

Nagy, et al. [7] proposed the CUDA Visual Library (VILIB). VILIB applies efficient low-level, GPU hardware-specific instructions to improve on existing computer vision algorithms in the field of VIO. It proposes a solution to

non-maxima suppression for feature detection on GPUs, and provides both Harris feature and fast feature detection algorithms. In terms of feature tracking, VILIB provides LK feature trackers. The VILIB feature detectors and trackers are tested using EuRoC datasets [11] and compared the execution time with other public feature detectors. The results indicate that the VILIB feature detectors and trackers achieve superior execution times compared to other CPU versioned equivalents. The front-end built with VILIB is integrated with existing VIO algorithms. The results show that VILIB accelerates the execution time. However, the accuracy is reduced, especially when fast camera motion and dark scenes occur.

C. Benchmark Comparison of VIO

Numerous studies have been conducted on the benchmarking of VO or VIO methods. The EuRoC datasets [11] are commonly used in different VIO works, such as [1], [5], and [7]. These datasets contain stereo images, synchronized IMU measurements, and accurate motion and structure ground-truth. They can be used to test both mono and stereo VIOs. EuRoC datasets provides three difficulty levels and three different scenes. Such diversity allows researchers to test VIO algorithms in different environments and motions. The rpg_trajectory_evaluation repository [12] implements common used trajectory evaluation methods for VO/VIOs. This repository supports both single and multiple trajectory estimation, as well as comparison between different algorithms on many datasets, including EuRoC MAV Datasets.

III. METHODOLOGY

The fundamental approach is to substitute the modules that consumes essential amount of CPU and computational time with corresponding modules enhanced by GPUs. These modules should capture the property that a huge amount of computation can be carried out separately, so that they can benefit from the parallel computing provided by GPUs.

We select VINS-mono as our baseline to build the new front-end. The front-end has large data volume to process, which contributes to high latency and CPU usage. We would like to balance the feature matching performance and speed. In addition, we are seeking for opportunities to extend the single camera system to multi-camera system. In multi-camera system, the CPU usage and latency will accumulate proportion to the number of cameras, making the VIO cannot achieve real-time processing. Among the front-end, the two key modules-feature detectors and feature trackers-are the most vital and time-consuming. However, if they can be solved in a parallel way, the cost will be reduced significantly. In VINS-mono, these two modules are achieved directly using functions provided in OpenCV [13]. In the OpenCV fashion, feature detection and tracking are implemented on CPU. Therefore, they are substituted with the corresponding GPUs implemented algorithms in our work.

In addition, we also try to implement the RANSAC algorithm on GPU and incorporate it into the VIO, since

the work of finding fundamental matrix and evaluating the estimated models can be done in a parallel way.

A. Incorporating VPI into the VIO

One major property of VPI is to support easy interoperation with existing projects that make use of OpenCV libraries. Therefore, the feature detection and tracking modules are directly substituted with corresponding VPI equivalents (Harris Corner Detector and Pyramidal LK Optical Flow).

We modify the underlying data structure in VINS-mono front-end to reduce the cost on data transportation between CPU and GPU. However, if the Harris features are picked according to their scores, their distribution is not uniform on the image. This might reduce the robustness in initializing the VIO. Inspired by ORB-SLAM [3], we implemented the quad-tree algorithm to make the selected feature points distribute more evenly inside the fisheye mask. The overall architecture of the front-end is shown in Fig. 1. The Design of quad-tree algorithm and result are shown in Algorithm 1 and Fig. 2, respectively.

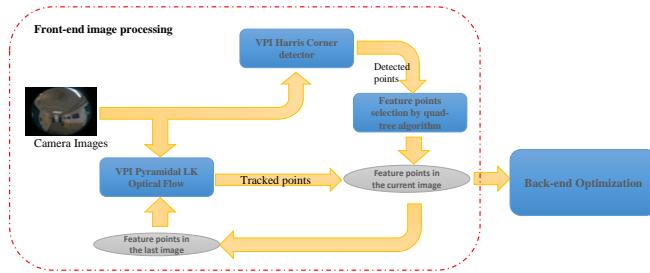


Fig. 1. The overall front-end architecture of VINS-mono with VPI.

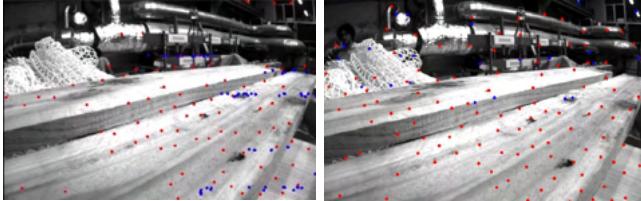


Fig. 2. Distribution of detected feature points. Here we use the same scene in MH_03_Medium of EuRoC datasets. In the picture, the red points are detected and tracked points and the blue points are newly detected points. The first picture shows the distribution of feature points detected by VPI without quad-tree algorithm. The feature points are simple selected by their scores. The second picture shows the detection results of VPI improved by quad-tree algorithm. It has more uniform distribution compared to the first picture.

B. Incorporating VILIB into the VIO

Similar to VPI, we replace the corresponding modules for feature detection and tracking with the modules in VILIB. However, the design of VILIB is not compatible with VINS-mono. We modify VILIB's structure, and encapsulate the algorithms for feature detection and tracking into functions having similar input and output format as that in OpenCV.

Algorithm 1 Selecting Feature points by Quad-tree.

Input: P_{in} : the set of detected feature points; S : the set of scores corresponding to detected feature points; k : the number of needed feature points; M : mask of fisheye.

Output: P_{out} : the set of selected feature points;

- 1: Select the feature points inside M into set P'_{in} ;
- 2: **if** $|P'_{in}| < n$ **then**
- 3: **return** P'_{in} ;
- 4: **end if**
- 5: Initialize image node set N , first image node n_1 ;
- 6: Store all feature points in P'_{in} into n_1 ;
- 7: Add n_1 into N ;
- 8: **while** $|N| < k$ **do**
- 9: **for** $n_i \in N$ **do**
- 10: **if** n_i has no feature point **then**
- 11: Remove n_i from N
- 12: **end if**
- 13: **if** n_i has at least 2 feature point **then**
- 14: Split n_i to four sub-nodes with the same image size;
- 15: **end if**
- 16: **end for**
- 17: **end while**
- 18: **for** n_i in N **do**
- 19: Select the feature point p_i having the highest score;
- 20: Add p_i to P_{out} ;
- 21: **end for**
- 22: **return** P_{out} ;

Among the feature detection algorithms provided by VILIB, we test the Harris corner detector and Fast feature detector.

During the testing, we found that the feature points tracked by VILIB LK tracker have serious drifting, making it impossible for VINS-Mono to do state estimation. Therefore, we only include the feature detectors of VILIB, while keep the CPU version of feature tracker.

C. RANSAC on GPU

The RANSAC algorithm is applied in the front-end to filter out mis-matched feature points. The basic procedure of RANSAC in VIO have three steps:

- 1) Randomly select eight pairs of points to compute fundamental matrices using the eight-point-algorithm [14,15];
- 2) Evaluate the matrices on all point pairs;
- 3) Pick the fundamental matrix which reports the most number of matched point pairs when termination requirement is satisfied.

The first two steps can be evaluated in a parallel way. Therefore, we define two kernels on CUDA to perform the first two steps, as shown in Fig. 3.

1) *Computing fundamental matrix*: Each GPU thread will compute one fundamental matrix. There are in total N threads, where N is the number of fundamental matrices we want to compute in each iteration. In our algorithm, N is set to 1024, since it is enough to finish RANSAC in one

iteration. Each GPU block contains p threads, where p is the number of tracked point pairs. In this way, the *shared memory* of GPU can be utilized to accelerate speed when fetching points data. Inside each kernel, it will randomly select eight pairs of points. Use the selected pairs, the eight-point-algorithm is performed to compute the fundamental matrix. Finally, the result is uploaded to a global array for later use.

2) *Evaluating models*: Each block will evaluate one model obtained in the previous step, and each thread inside the block will compute the score of one pair of point. Therefore, the number of blocks and threads are N and Np , respectively. Similarly, the *shared memory* is used in each block to accelerate data transfer speed. If the score is lower than a certain threshold, then the point pair specified by the thread is considered as an inlier, and the thread uses the atomicAdd function to increase the score of the model by 1.

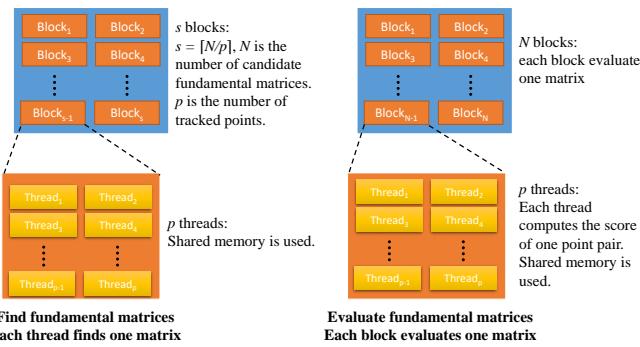


Fig. 3. The implementation of kernels of finding fundamental matrices and evaluating the matrices.

IV. RESULTS

We test VINS-mono with GPU enhanced algorithm on EuRoC datasets [11]. The CPU for testing is *AMD Ryzen 5800H with Radeon Graphics 3.20GHz*. For each dataset, we conduct five trials for both VINS-Mono and VINS-Mono with GPU enhanced algorithms. Then we use the modules provided in rpg_trajectory_repository [12] to evaluate and compare the results. In this section, the CPU usage and computation latency of different VIO front-ends are also compared. We also collect our own data use our own VIO setup with fisheye cameras and evaluate the performance. The setup is shown in Fig. 4.

A. Performance of RANSAC on GPU

We compare our RANSAC on GPU with the RANSAC algorithm provided by OpenCV. The feature points are data collected from VINS-mono. The result is shown in the table I and II.

Table I shows that RANSAC on GPU is slower than RANSAC on CPU, indicating that the GPU version of RANSAC cannot accelerate the processing rate. According to Table II, it is noticed that copying data from CPU to GPU and releasing memory on GPU and CPU consumes 65% of



Fig. 4. Our own VIO setup. From left to right, the devices are: battery, helmet with fisheye cameras and IMU, Xavier.

time. Since the size of feature points collected in VINS-mono is limited to a relatively small amount, copying data between CPU and GPU becomes significant in the entire RANSAC process. With small size of feature points, CPU versioned RANSAC algorithm can operate in a fast speed. Therefore, the total speed of RANSAC on GPU is slower than that on CPU, even though finding and evaluating fundamental matrices, which is the major body of RANSAC algorithm, saves half of time for RANSAC on GPU. On the other hand, compared to the feature detection and tracing, the scale of time for RANSAC is relatively small. Improving it has few effect on the performance of the VIO. Considering these two reasons, we do not include RANSAC on GPU in the following testing.

TABLE I
COMPARISON OF RANSAC ON GPU AND CPU

RANSAC on CPU	RANSAC on GPU
0.169ms	0.352ms

TABLE II
DETAILED TIME CONSUMPTION OF RANSAC ON GPU

Copying data from CPU to GPU	0.159ms
Finding and evaluating fundamental matrices	0.085ms
Releasing memory on GPU and CPU	0.071ms

B. Comparison of Different versions of VINS-mono

Table III and IV shows the CPU usage and computational latency of different versions of VINS-mono. We also test VINS-mono with VPI on NVIDIA Xavier AGX. In the experiments, the VIO setup starts from the same position, travels through the same trajectory and finally comes back to the starting position. The testing result is shown in Fig. 5.

Table III shows VINS-mono-VPI has the most significant reduce on CPU usage and computational latency, achieving

TABLE III

CPU USAGE (FEATURE DETECTION + TRACKING) AND COMPUTATIONAL LATENCY FOR DIFFERENT VERSION OF VINS-MONO

Version	CPU usage (single core)	Computational latency (feature detection + tracking)
VINS-mono	52%	15.4ms
VINS-mono-VPI	31%	7.8 ms
VINS-mono-VILIB fast feature	49%	8.5ms
VINS-mono-VILIB Harris feature	51%	8.7ms

TABLE IV

CPU USAGE (FEATURE DETECTION + TRACKING) AND COMPUTATIONAL LATENCY FOR DIFFERENT VERSION OF VINS-MONO WITH FISHEYE CAMERA

Version	CPU usage (single core)	Computational latency (feature detection + tracking)
VINS-mono	85%	20.6ms
VINS-mono-VPI	53%	11.7ms
VINS-mono-VILIB fast feature	82%	28.7ms
VINS-mono-VILIB Harris feature	91%	16.2ms

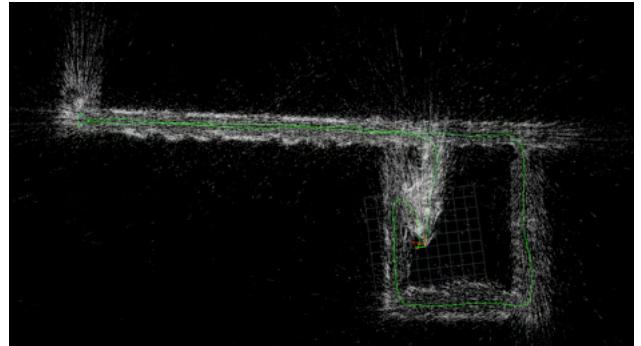
a reduction of 40.4% and 50.6%, respectively. Modules from VILIB do reduce the computation latency, but they do not reduce the CPU usage too much. According to Table IV, when fish eye is used, all the CPU usage and computation latency increase. This results from additional work on undistorting feature points. It is noticed that excepted VINS-mono with VPI, all the other three versions have single core CPU usages close to 100%. For VINS-mono with faster feature detector of VILIB, the computational latency is even greater than that of VINS-mono.

According to the results on Xavier, the CPU usage on both front-end and back-end is reduced for VINS-mono with VPI. The reason that the CPU usage on back-end is reduced is because the number of tracked feature points in consecutive frames is lower. Therefore, points used in optimization is smaller. The accuracy is maintained according to the estimated trajectories.

Fig. 6 and Fig. 7 shows the trajectory evaluation results. They present the translation RMSE and rotational RMSE on different versions of VINS-mono. VINS-mono with VPI has similar accuracy as that of VINS-mono. Sometimes it is better than VINS-mono. On the other hand, VINS-mono with VILIB lose accuracy to some extend, especially for fast feature detectors when testing datasets V1_02_medium, and V2_02_medium. Fig. 8 privides an example evaluation from V1_02_medium. VINS-mono, VINS-mono-VPI, and VINS-VILIB-harris have relatively small drift from the groundtruth. VINS-VILIB-fast has a conspicuous drift from the groudtruth.

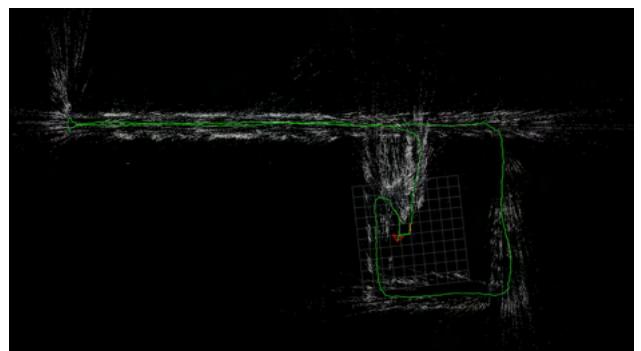
C. Discussion

VPI enhanced VINS-mono both maintains the accuracy of VIO, and reduces the CPU usage and computational latency



VINS-mono without VPI, Shi-Tomasi Feature

Thread	CPU usage
Feature tracking	70% 1 core
Optimization	60% 1 core



VINS-mono with VPI, Harris Feature

Thread	CPU usage
Feature tracking	37% 1 core
Optimization	36% 1 core

Fig. 5. Comparison of CPU usage (single core) of different versions of VINS-mono on Xavier. The first group is the result obtained from original VINS-mono. The feature type is Shi-tomasi feature in the original VINS-mono. The second group is the result obtained from VINS-mono with VPI. The table below each picture shows the CPU usage on the front-end (feature tracking) and back-end (optimization). The green lines in the pictures are the estimated trajectories. The white dots around them are tracked feature points cloud. The red dot indicate the start position of the estimation.

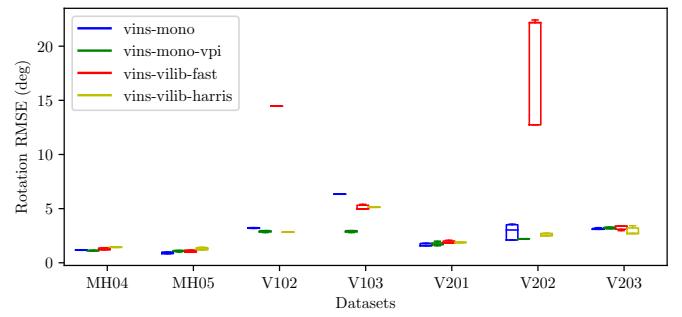


Fig. 6. Rotational RMSE for different versions of VINS-mono. The dataset in EuRoC are MH.04_difficult, MH.05_difficult, V1_02_medium, V1_03_difficult, V2_01_easy, V2_02_medium, V2_03_difficult.

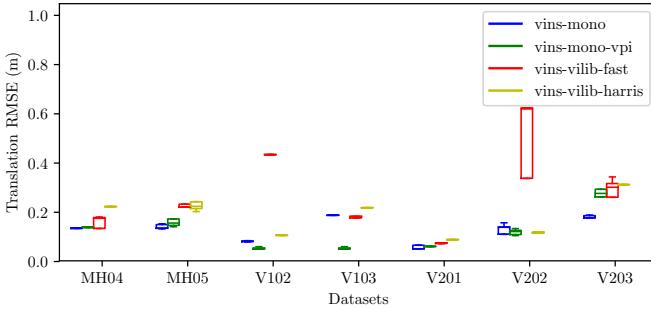


Fig. 7. Translational RMSE for different versions of VINS-mono. The dataset in EuRoC are MH_04_difficult, MH_05_difficult, V1_02_medium, V1_03_difficult, V2_01_easy, V2_02_medium, V2_03_difficult.

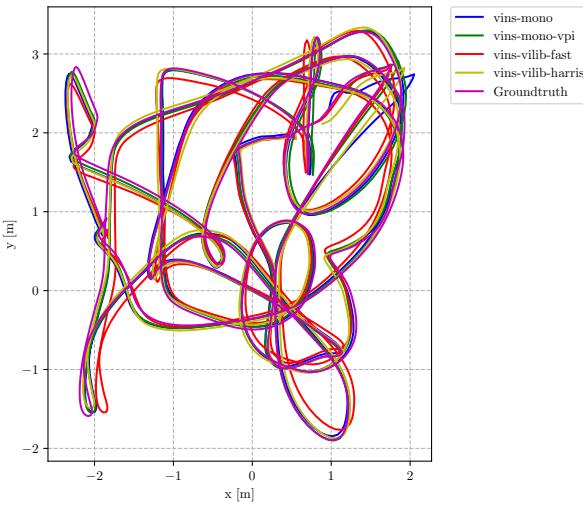


Fig. 8. Evaluated trajectories of different versions of VINS-mono on V1_02_medium. While VINS-mono, VINS-mono-VPI, VINS-VILIB-harris have small drift from the groundtruth, VINS-VILIB has a conspicuous drift from the groundtruth.

significantly. It reduces the CPU usage and computation latency by 40.4% and 50.6%, respectively. VILIB reduces the computation latency to some extend, but it does not reduce the CPU usage. It also has larger estimation errors compared to VINS-mono and VINS-mono with VPI. Therefore, we choose VPI to enhance the performance of the VIO, and use it to build the new front-end.

V. CONCLUSIONS

In this paper, we present the GPU-enhanced front-end for VINS-mono. The results show VINS-mono with VPI reduces CPU usage and computation latency significantly, while the state estimation accuracy maintains. The results also indicate that the modules provided by VILIB and RANSAC on GPU cannot reduce CPU usage and computation latency in VINS-mono.

In the future, we would like to extend the mono-VIO to multi-camera VIO, where each camera has little FoV overlap. Comparing to single camera VIOs, such multi-camera VIO will fully utilize the advantages brought by

VPI, achieving a real-time and low-cost state estimation. In addition, we would like to incorporate the gstreamer pipeline into VINS-mono with VPI. There are several reasons for using it. First, VPI functions as a plugin in gstreamer. Second, gstreamer allows to transfer multiple image streams simultaneously without causing extra cost on CPU. With gstreamer, we would like to further reduce the cost on CPU and computation latency, especially for multi-camera VIOs.

VI. ACKNOWLEDGEMENT

This work is supported by the Robotics Institute Summer Scholars Program (RISS), Carnegie Mellon University, the AirLab and the Chinese University of Hongkong, Shenzhen.

REFERENCES

- [1] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, 2018.
- [2] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filer based visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, vol. 33, no. 10, pp. 1255–1262, 2017.
- [3] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.
- [4] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
- [5] J. Jeon, S. Jung, E. Lee, D. Choi, and H. Myung, "Run your visual-inertial odometry on nvidia jetson: Benchmark tests on a micro aerial vehicle," *arXiv preprint arXiv:2103.01655*, 2021.
- [6] NVIDIA. (2021) Vpi-vision programming interface documentation. Accessed: 2021-07-28. [Online]. Available: <https://docs.nvidia.com/vpi/>
- [7] B. Nagy, P. Fohnen, and D. Scaramuzza, "Faster than FAST: GPU-accelerated frontend for high-speed VIO," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2020.
- [8] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [9] H. Strasdat, J. M. Montiel, and A. Davison, "Visual slam: Why filter?" *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.
- [10] CUDA. (2021) Programming guide::cuda toolkit documentation. Accessed: 2021-07-28. [Online]. Available: <https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html>
- [11] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achterlik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, 2016. [Online]. Available: <http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract>
- [12] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.
- [13] OpenCV. (2021) Open source computer vision library. Accessed: 2021-07-28. [Online]. Available: <https://github.com/opencv/opencv>
- [14] R. I. Hartley, "In defense of the eight-point algorithm," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 6, pp. 580–593, 1997.
- [15] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pp. 61–62, 1987.

AlgeGloves: An Interactive Algebra Interface That Allows Students to Mold Algebraic Functions

Iqui Balam Heredia Marin¹ and Melisa Orta Martinez²

Abstract—Math is often presented as an abstract, procedural discipline in which both teacher and student have a firm belief that proficiency is due to fixed innate ability. English learners and minority students are especially vulnerable to these misconceptions as they often face further discrimination and fewer resources and support. However, evidence has shown that most students are capable of excelling in and enjoying math, but not enough interactive tools are provided for them. In this paper we present AlgeGloves, an interactive software interface that elevates the understanding of mathematics language by providing embodied non-verbal representations of math concepts. We have developed a software interface which allows students to center mathematics around exploration. Using a color detection algorithm coupled with the user wearing colored gloves, we can detect users' movements and allow them to manipulate mathematical functions as if they were manipulating clay. Results show an intuitive interface, that can be controlled to solve proposed exercises, experimenting with algebra through hand movement.

Index Terms—Computer Vision, Algebra, Interface, Interactive, Education

I. INTRODUCTION

A. Motivation

Even though mathematics is considered an abstract subject, research has shown that abstract mathematics concepts can be connected and grounded through bodily interactions [1] which has sparked the development of a wide range of dynamic and interactive environments to teach mathematics [2]. Research has also shown that a lack of interactive experiences with mathematics concepts could lead to a gap in experimentation for students. These studies have suggested that kinesthetic lessons, incorporating movement in the process of learning instead of just listening and visualization, have been associated with better student outcomes [3]. Algebra, in particular, is considered the gateway to abstract thought [4]. In this work we present AlgeGloves, an interactive software algebra interface that allows students to experiment and interact with algebraic functions through movement. We propose that the enhanced interactivity of the learning interface will have positive effects in learning outcomes as students will be able to ground abstract mathematics concepts in bodily movement.

B. Prior Work

Prior research has shown the potential of learning through gestures and actions. In [5] for example, researchers showed

that both physical actions with objects as well as gestures helped students solve problems in which they were trained. They also observed that gestures led to improved student outcomes in solving problems which required knowledge transfer.

Researchers have also explored the potential of games which focus on teaching students mathematical skills through different interactions with a virtual environment. Some of these games aim to teach mathematics by immersing students in real-world familiar environments situations [6]. The work by Castellar et al. [7] showed that in game- and interactive-based learning students have more positive affective responses than when they are learning through traditional paper methods. Other games use Virtual Reality (VR) in order to immerse the learner in a world where they can manipulate objects and explore the simulated world as they are learning mathematical concepts [8]–[10].

A recent study reported benefits in adding kinesthetic interactions to learning environments [11]. In this work, researchers developed a kinesthetic interface to teach physics concepts such as Position, Velocity, and Acceleration by having students draw the graph using their hands movement, a Kinect which could detect these movements, and a computer which would draw on the screen the graphs that were created by the student's hand movements.

II. ALGEGLOVES: AN INTERACTIVE ALGEBRA INTERFACE

One of the most important goals when developing AlgeGloves, was to create an interactive interface that could be used by teachers and students in K-12 classrooms. Thus, we opted for creating an interactive interface that could run on a laptop using the laptop camera. We focused our work on Algebra and Algebraic functions since these topics are considered gateways to abstract thought [4] and there are already several computer interfaces that could potentially incorporate this technology [2].

In order to use AlgeGloves, the students wear a pair of gloves (shown in Fig. 2). These have three small colored circles each in one finger: thumb, index and middle finger. These fingers were selected since they are the most used when manipulating and pinching objects. Using these gloves students can then manipulate a clay-like material shown on the computer screen in order to create various mathematical functions.

The workflow of the AlgeGloves computer interface is described in Fig. 1 and is divided into two components: The first is in charge of processing the incoming image from

¹Iqui Balam Heredia Marin is with the Instituto Tecnológico y de Estudios Superiores de Monterrey, iquibalamhm@gmail.com

²Melisa Orta Martinez is with Carnegie Mellon University, mortamar@andrew.cmu.edu

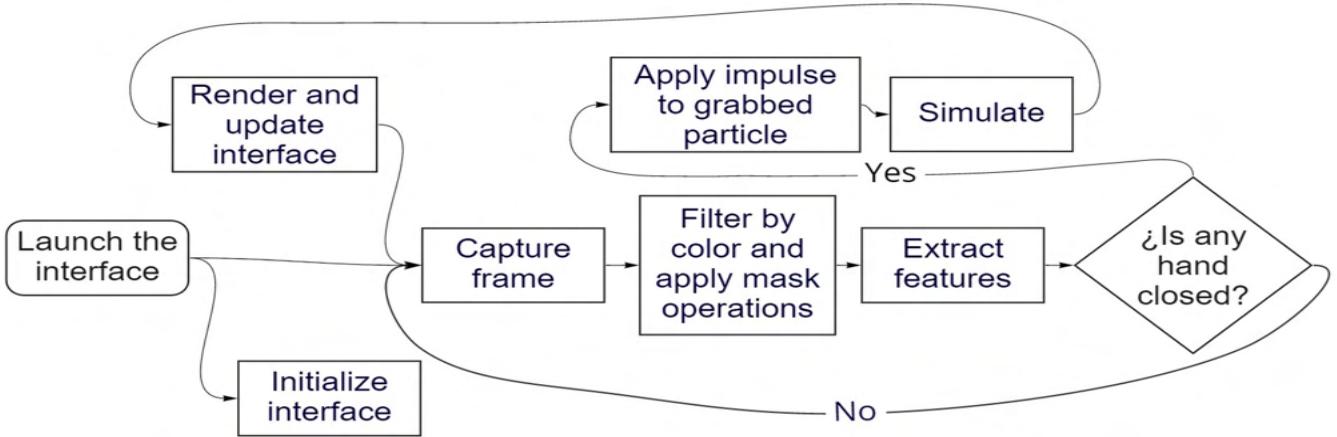


Fig. 1. Workflow of operation for the interface



Fig. 2. Gloves used for interaction with AlgeGloves. Students wear a pair of gloves which have three colored circles each one in one finger: thumb, index and middle finger.

the computer camera and extracting the user's motion from it. The second is in charge of rendering the clay which is composed of a cluster of connected particles. In order to render the clay-like behavior of the particles in a realistic way, a physics engine updates their position based on the user's interactions with the clay.

When a user first launches AlgeGloves, it initializes the particle start positions. These can be set by an instructor depending on the function that they would like students to mold. During this initialization step, the computer camera will also capture a frame and perform several filtering layers (described below) in order to extract the three colored dots that are placed on the gloves that the user wears on each hand.

After extracting the three dots which represent the user's hand position, AlgeGloves then uses the center point of those dots to decide if any of the hands are performing a "pinching"

motion or not. If the hands are not closing, or closed, no pinching motion is detected and the program proceeds to capture more frames. If the hands are pinching the clay, the program then applies an impulse force to the particles that the user is interacting with and updates the position of all the connected particles using the Verlet Integration method described in Section III.

When a user stops interacting with the particles, they will hold the position that the user moved them to and the program will go back to capturing frames until the user "pinches" the clay again or exits the interface.

AlgeGloves allows the user to reset the particle positions to their initial state, so that all the modifications done to that point are removed and they may be able to start their current exercises. In order to do so, we implemented a *closing gesture* in which users cross their hands and then close them (Fig 3).

To close AlgeGloves interface the user may press the *Esc* key or just close the window with the common cross at the right upper corner.



Fig. 3. Reset Gesture: Both hands have to be closed but in their opposite side

The aim of AlgeGloves is to create an affordable interface which allows students to explore math by simply putting on a pair of gloves. Thanks to the calibration algorithms

(explained in Section III), the AlgeGloves interface does not have to be deployed on a specific device. This way learning can happen through manipulation and motion instead of on paper. Our goal is to have students map math concepts to bodily movements and to incite a deeper interest in mathematics, and better learning outcomes.

III. METHODOLOGY

In this section, the different parts of the simulation are explained. The physics simulation that governs the particles movement, and the computer vision algorithm that processes the input frames and decides what to do with the given information.

A. Physics Simulation

The purpose of the physics simulation is to render the clay that the students will manipulate. This is done by rendering hundreds of individual particles which are connected to each other through a spring-damper system. At each time step, the particles position is updated based on their previous dynamics and new forces being applied to them. In order to render this behavior in a way that mimics a soft body such as clay, we use the Verlet Integration Method.

Rigid body physics is well understood, but rendering soft body physics is not completely covered in literature. One approach to rendering soft bodies is through the Verlet Integration Method, developed by Loup Verlet to solve Newton's equations of motion. This method is often used when modeling the physics of deformable bodies. The Verlet Integration Method can be implemented along with a visualizer as in [12] and its integration solution for position is shown in Equation 1:

$$\mathbf{x}(n+1) = 2\mathbf{x}(n) - \mathbf{x}(n-1) + \mathbf{a}(n) * (\Delta t) \quad (1)$$

where:

- a is the acceleration of the system.
- x is the position of the particles.
- Δt is the timestep for the simulation.

The Verlet integration method is stable and it can be implemented using classes and objects in any object oriented programming language. Using this method we can update the particles' position through impulses resulting from forces applied by the user.

B. Computer Vision Algorithm

The computer Vision algorithm's main goal is to detect and track the user's hands motions as they are interacting with AlgeGloves. It is based on three main steps: first we calibrate the color boundaries, then we filter based on those boundaries, and finally we decide what to do with the final mask. In this section we describe those steps. The full algorithm was coded in Python 3.8 and uses OpenCV version 4.5.1.

1) Calibration: A simple script allows the user to identify the colors to single out by setting the Hue, Saturation, and Value (HSV) parameters using slider bars. The algorithm then filters the background of the video image out using a pass-through filter. Fig. 4 shows an example of this process being carried out: A green piece of paper in the image is singled out (Fig. 4A), and converted into a clean binary mask (Fig. 4B).

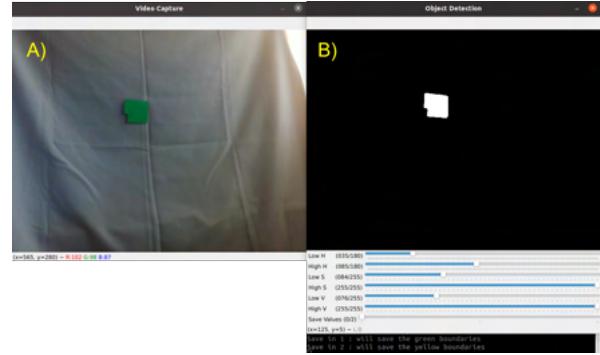


Fig. 4. Calibration Stage. A) Input image without any filtering, B) Image after a simple HSV filter for calibration has been applied

2) Filtering: In most cases, a single pass-trough filter is not enough to obtain a clean mask. Fig. 5A shows the mask output when a single pass trough filter is applied in non-favorable conditions. Fig. 5B shows the output when a more elaborated processing stage is used.

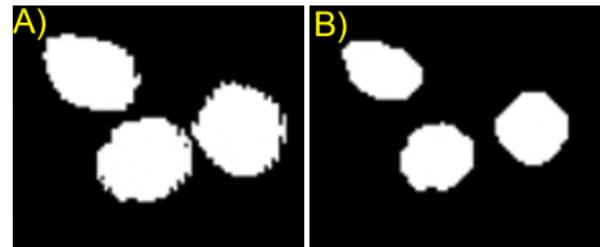


Fig. 5. Filter stage. A) HSV filtered image in non-favourable conditions, B) Mask image after applying an elaborated filtering stage

This filtering stage is executed as follows.

- 1) A pass-trough filter using the calibrated lower and higher HSV boundaries.
- 2) Morphological open operation, to close tiny holes that may exist, using an elliptic structuring element with a dimension of (3,3) and 3 iterations.
- 3) Get contours and areas of current blobs, the areas with an area of less than 70 pixels are discarded.
- 4) At this point, the image is not binary so, a threshold operation is used to either write pixel color as 0 or 255.

This filter not only removes small white dots but makes sure that the fingers are correctly spaced and not combined.

3) Decision Maker: The mask obtained after the filtering stage consists of three contours. Using these contours we calculate the center position of the hand. The algorithm is as follows:

- 1) Obtain the center of each of the three areas.
- 2) Compute the area of the triangle with vertexes in this three centers.
- 3) If the area is less than a threshold of 1200 pixels, we state that the hand is Closed, otherwise it is Open.

To find the point of interest in the hand we define the barycenter, or centroid, of the hand triangle, as follows:

$$\text{Centroid}\left(\frac{x_1 + x_2 + x_3}{3}, \frac{y_1 + y_2 + y_3}{3}\right) \quad (2)$$

The centroid of the hand, is the point that AlgeGloves uses as a user's cursor. Using this cursor, the user can manipulate mathematical functions, but only when the hand is closed.

IV. RESULTS

Once the image processing and the physics were designed, the visualization and interaction environment was built using pygame (version 2.0.1), a tool for creating videogames with Python. To test the correctness of the physics model and the interaction with the gloves, two different exercises were created. In these activities, the goal for the student is to match a given equation with the actual equation of the manipulated clay. Once we have an error of 5% the task is complete and the interface will show a different Target equation pre-selected by a instructor / professor. Figures 6 and 7 show these two activities. During the activities, the user manipulates the white line which behaves like a clay while the red line shows the fitted mathematical function to that clay.

For the manipulation, both gloves can be used alternately, and the colors of the gloves can also be different to the presented in Fig. 2. In order to use other colored gloves, the user needs to calibrate those colors using the process described in the Calibration Stage (Fig. 4).

In the following subsections we describe the two activities developed.

A. Straight Line

The purpose of this activity is to have users develop an understanding of the linear equation and the meaning of the slope of an equation. In order to do this, users are tasked with matching a linear equation which appears on the left upper corner (the *Target* equation) by manipulating the white particles on the screen. Users can observe on the right top corner of the screen a fitted equation to the clay they are manipulating as well as a red line which is super-imposed on the clay. This equation is fitted using a *scipy* optimizer, a Python library focused on machine learning techniques. In order for the particles to be constrained to linear-like functions, Fig. 6, the clay is constrained at the center particle. The user can then manipulate the function using the left and right-side particles as they pinch or close their hands and rotate this line function. The concept of slope then becomes a rotating gesture which users perform as they are moving a linear function to match the target function.

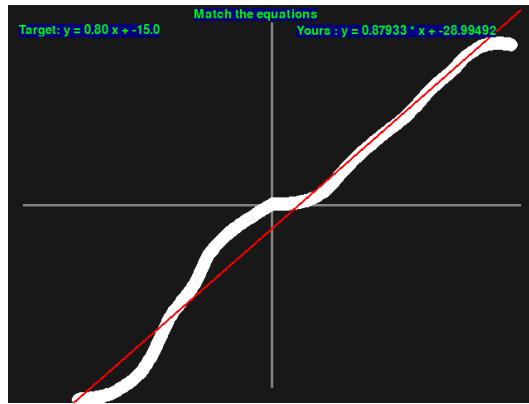


Fig. 6. Interactive screen for Straight Line interaction

B. Parabola

As shown in Fig. 7, the parabola is constrained at the left and right ends, and users are free to manipulate all the inner particles. Similar to the Straight Line, there is a *Target* equation and the user's fitted equation equation, but now approximated using a second grade polynomial optimizer of *scipy*. The approximated equation is again displayed in red. The purpose of this simulation is for users to develop an understanding of a squared term and rate of growth in an equation through angled hand movement.

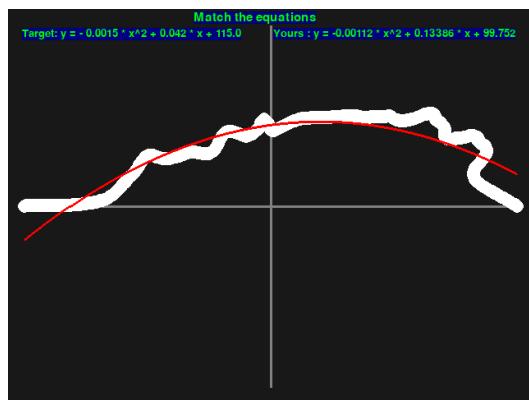


Fig. 7. Interactive screen for Parabola interaction

V. CONCLUSIONS

Current work establishes the foundation of a working algebra interface where students can interact with mathematical functions. Future efforts will consider the relevance and differentiation between this kinesthetic way of learning against the usual and abstract teaching. In future work we will also enhance the interface by adding haptic feedback to the gloves. This will allow the users to feel the functions as they are manipulating them and encourage movement. Another important future goal is to further develop this interface as well as the haptic feedback in partnership with teachers learning scientists.

VI. ACKNOWLEDGMENTS

This work is supported by the School of Computer Science (SCS), Carnegie Mellon University. Special thanks to Dr. John Dolan and Rachel Burcin for making RISS 2021, completely remote, possible.

REFERENCES

- [1] G. Lakoff and R. Núñez, *Where mathematics comes from*. New York: Basic Books, 2000, vol. 6.
- [2] M. Cevikbas and G. Kaiser, “A systematic review on task design in dynamic and interactive mathematics learning environments (DIMLEs),” *Mathematics*, vol. 9, no. 4, p. 399, feb 2021.
- [3] C. R. BEAUDOIN and P. JOHNSTON, “The impact of purposeful movement in algebra instruction.” *Education*, vol. 132, no. 1, pp. 82 – 96, 2011. [Online]. Available: <http://0-search.ebscohost.com.bibliotecas.tec.mx/login.aspx?direct=true&db=asxAN=66538814&lang=essite=eds-livescope=site>
- [4] B. S. Witzel, “Using cra to teach algebra to students with math difficulties in inclusive settings.” *Learning Disabilities: A Contemporary Journal*, vol. 3, no. 2, pp. 49–60, 2005.
- [5] M. A. Novack, E. L. Congdon, N. Hemani-Lopez, and S. Goldin-Meadow, “From action to abstraction: Using the hands to learn math,” *Psychological Science*, vol. 25, no. 4, pp. 903–910, 2014, PMID: 24503873. [Online]. Available: <https://doi.org/10.1177/0956797613518351>
- [6] C. H. Muntean, N. El Mawas, M. Bradford, and P. Pathak, “Investigating the impact of an immersive computer-based math game on the learning process of undergraduate students,” in *2018 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2018, pp. 1–8.
- [7] E. N. Castellar, A. All, L. De Marez, and J. Van Looy, “Cognitive abilities, digital games and arithmetic performance enhancement: A study comparing the effects of a math game and paper exercises,” *Computers & Education*, vol. 85, pp. 123–133, 2015.
- [8] X. Xu and F. Ke, “Designing a virtual-reality-based, gamelike math learning environment,” *American Journal of Distance Education*, vol. 30, no. 1, pp. 27–38, 2016.
- [9] M. Simonetti, D. Perri, N. Amato, and O. Gervasi, “Teaching math with the help of virtual reality,” in *International Conference on Computational Science and Its Applications*. Springer, 2020, pp. 799–809.
- [10] S. Putman and L. Id-Deen, ““ i can see it!”: Math understanding through virtual reality.” *Educational Leadership*, vol. 76, no. 5, pp. 36–40, 2019.
- [11] N. A. R. Ayala, E. G. Mendivil, P. Salinas, and H. Rios, “Kinesthetic learning applied to mathematics using kinect,” *Procedia Computer Science*, vol. 25, pp. 131–135, 2013, 2013 International Conference on Virtual and Augmented Reality in Education. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1877050913012210>
- [12] D. Ramtal and A. Dobre, “Numerical integration schemes,” in *The Essential Guide to Physics for Flash Games, Animation, and Simulations*. Springer, 2011, pp. 443–459.

Delay-aware Control for Safe Autonomous Driving

Dvij Kalaria¹, Qin Lin², and John M. Dolan²

Abstract—With the advancement of affordable self-driving vehicles with limited computation resources and complicated nonlinear optimization, computation time becomes a matter of concern. Other factors such as actuator dynamics and actuator command processing delay also unavoidably cause delays before the planned command influences the vehicle dynamics. In high-speed scenarios, these delays are critical to the safety of a vehicle. Recent works consider each of these delays individually, but none of them considers all of them under one approach in the context of autonomous driving. Moreover, recent works inappropriately consider computation time to be a constant or as an upper bound, which makes the control either less responsive or over-conservative. This paper, however, presents a unified framework by modelling actuation dynamics, using robust tube model predictive control and a Kalman filter to deal with all these delays, which makes the controller safe while minimizing conservativeness. On one hand, our approach can serve as a standalone controller; on the other hand, our approach can provide a safety guard by a low-level refinement of the control commands obtained from the high-level controller, which assumes no delay. This can be used for compensating the sim-to-real gap when deploying a black-box learning-enabled controller trained in a simplistic environment without considering delays for practical vehicle systems.

Index Terms—Autonomous driving, Delay compensation, Actuator dynamics, Computation delay

I. INTRODUCTION

The recent surge in autonomous driving cars has led to an inevitable increase in demand for making them more affordable and accessible. The stringent requirements of on-board computing and high-resolution sensors pose a major challenge in making self-driving cars affordable. While there has been much work on making current algorithms more efficient by reducing algorithm complexity and parallelizing, the required computation time may still be dangerous for high-speed scenarios like highways, where rapid action may be needed in case of an anomaly to avoid an accident.

Due to highly sophisticated algorithms required to control the vehicle, the planned command takes time to be processed by the actuator before it gets executed. Most commonly used control algorithms assume the planned steering command to be applied instantaneously. But due to dynamic constraints, this assumption can generate significant tracking errors and jeopardize stability, especially in high-speed scenarios. This work presents a unified approach to dealing with three types of delay: 1) computation time delay; 2) actuator command processing delay; and 3) actuator dynamics delay. First, we model the actuator dynamics delay using a first-order ordinary differential equation (ODE). Second, we propose a delay-aware robust tube Model Predictive Control (MPC) coupled with a Kalman filter to probabilistically safely estimate the computation time delay. We propose two controller plans. The first plan enables our approach to serve as a standalone controller for a delay-aware robust control. In the second controller plan, there exists a primary controller in a closed-loop

system without considering delays. Our approach can compensate the primary controller to boost its safety performance. This controller plan has application benefits, since learning-enabled (LE) controllers are being used in autonomous systems. However, simplistic assumptions are usually made in the training procedure of the LE controllers. For safety-critical autonomous systems such as autonomous driving cars, it is crucial to close the sim-to-real gap by providing a safety-assured guard. We treat the LE controller as a high-level controller. In the low-level control part, we track the reference generated by the LE controller, but actively regulate its unsafe control by compensation considering practical delays.

In summary, we make the following novel contributions:

1. A unified delay-aware robust control approach dealing with three major delays: computation time delay, actuator command processing delay, and actuator dynamics delay.
2. A probabilistic framework for estimation of computation time instead of taking an upper bound, which makes the algorithm safe while minimizing conservativeness.
3. A control plan to safely compensate for these delays for a LE controller which doesn't consider delays.

The rest of this paper is organized as follows. Section II provides a review of some important related work. Section III describes the methodology. Section IV presents the experimental results. The conclusions are in Section V.

II. RELATED WORKS

A popular architecture for layered planning and control involves a high-level planner and a low-level controller for trajectory tracking such as Ackermann controller [1], pure pursuit [2], Stanley [3], Model Predictive Control (MPC) [4], and preview controller [5] to name a few. In the low-level control layer, many algorithms ignore the delay arising from various factors such as computation, actuator command processing, sensor delay, and actuator dynamics. For the compensation of computation delay for discrete MPC, Ref. [6] proposes a simple solution of shifting the initial state by one control step, approximating it from the prediction model. However, this is not suitable if the computation time is more than one control step. Ref. [7] further proposes to use a buffer to store control commands from the previous batch. It also proposes the use of a pre-compensating unit and robust tube MPC to prove the safety of the system under bounded perturbation. This plan is well suited for static scenarios, where the objective is nearly constant throughout. However, for highly dynamic scenarios where the planned trajectory changes rapidly, taking the upper bound as the horizon length might lead to the algorithm's being less responsive. We instead propose to use a Kalman filter to approximate a local upper bound on computation time and

¹Dvij Kalaria is with the Department of Computer Science and Engineering, IIT Kharagpur, India dvij.kalaria@gmail.com

²Qin Lin and John M. Dolan are with the Robotics Institute, Carnegie Mellon University qinlin, jdolan@andrew.cmu.edu

adapt to its changing values instead of taking an upper bound as the horizon length.

For compensation of delay due to actuator dynamics, Ref. [8] proposes approximating the actuator dynamics by a first-order ODE, after which the actuator state can be augmented in the state space model. The approach has been tested on differential braking stability control. Instead, we use this similar design for compensating steering delay.

For considering delay caused due to processing of actuator commands at the actuator, [5] proposes an initial state transition method similar to the compensation for computation delay but for a preview continuous controller. It proposes a closed-loop solution to compensate for a dead time between when the command is planned and when it reaches the actuator. [9] further extends the idea by including compensation for actuator saturation as well to make the solution deployable on real systems with control limits. However, with the use of the preview controller, it becomes difficult to include state constraints in the system. [10] proposes a simple way to compensate for the sensor delay by transforming the frame of sensor values to reflect values at the current time and not at the time when they were recorded. Our work, however, considers computation delay, actuator command processing delay, and actuator dynamics as well as control and state constraints under one optimization framework in the context of autonomous vehicle control.

III. METHODOLOGY

A. Notation

A polytope is defined as a convex hull of finite points in d -dimensional space as \mathbb{R}^d . The Minkowski sum of two polytopes is defined as $P \oplus Q := \{x+q \in \mathbb{R}^d : x \in P, q \in Q\}$. The Pontryagin difference of two polytopes is defined as $P \ominus Q := \{x \in \mathbb{R}^d : x + q \in P, q \in Q\}$

B. System dynamics

A kinematic bicycle model describes the dynamics of the vehicle with the state variables being position (p_x, p_y), heading angle (θ), and velocity (v), and the control variables acceleration (a) and steering angle (δ). It is commonly assumed in the literature that steering angle δ and acceleration a are applied instantaneously by the actuators. But in reality there is a certain lag between the command and when the actuator physically modifies the steering angle state which is called the actuator dynamic delay. We modify the system dynamics to also include the actual steering angle as a state denoted as δ_a . The control command is now the desired steering angle δ . We approximate the change in the steering angle state by a first-order ODE similar to [8], i.e., $\dot{\delta}_a = K(\delta - \delta_a)$, where K is the inverse of the time constant for the steering actuator. For acceleration, pedal dynamics are assumed to be instantaneous for the experiments in this paper. However, they can also be approximated in the same way. After discretization, the vehicle state is now modified as $x_k = [p_{x,k}, p_{y,k}, \theta_k, v_k, \delta_{a,k}]$. The discrete dynamics are given in Equation 1.

$$p_{x,k+1} = p_{x,k} + \frac{\sin(\theta_k + \kappa_k l_k) - \sin(\theta_k)}{\kappa_k} \quad (1a)$$

$$p_{y,k+1} = p_{y,k} + \frac{\cos(\theta_k) - \cos(\theta_k + \kappa_k l_k)}{\kappa_k} \quad (1b)$$

$$\theta_{k+1} = \theta_k + \kappa_k l_k \quad (1c)$$

$$v_{k+1} = v_k + a_k \Delta t \quad (1d)$$

$$\delta_{a,k+1} = \delta_k - (\delta_k - \delta_{a,k})(e^{K \Delta t} - 1) \quad (1e)$$

$$(1f)$$

where the curvature $\kappa_k = \frac{\tan(\delta_{a,k}) C_r}{L}$, L is the vehicle length, and the travel distance $l_k = v_k \Delta t + \frac{1}{2} a_k \Delta t^2$.

C. Robust Tube MPC

For a discrete linear system with system matrices $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$, let the control gain $K \in \mathbb{R}^{m \times n}$ be such that the feedback system of $A_K = A + BK$ is stable. Let \mathcal{Z} be the disturbance-invariant set for the controlled uncertain system $x^+ = A_K x + w$, satisfying $A_K \mathcal{Z} \oplus \mathcal{W} \subseteq \mathcal{Z}$, where the disturbance w is assumed to be bounded ($w \in \mathcal{W}$) by a polyhedron that contains the origin in its interior. The following finite optimization problem is solved at each step for $\bar{X} = \{\bar{x}_0, \bar{x}_1, \dots, \bar{x}_N\}$, $\bar{U} = \{\bar{u}_0, \bar{u}_1, \dots, \bar{u}_{N-1}\}$ and reference state sequence $X_{ref} = \{x_{ref,0}, x_{ref,1}, \dots, x_{ref,N}\}$ obtained from the path planner, where N is the horizon length, and \bar{X} and \bar{U} are the state and control sequences of a nominal system ignoring w .

$$\begin{aligned} \min_{\bar{x}_0, \bar{U}} \quad & \sum_{t=0}^{N-1} (\bar{x}_k - x_{ref,k})^T Q (\bar{x}_k - x_{ref,k}) + \bar{u}_k^T R \bar{u}_k \\ & + (\bar{x}_N - x_{ref,N})^T Q_N (\bar{x}_N - x_{ref,N}) \\ \text{s.t.} \quad & \bar{x}_{k+1} = A \bar{x}_k + B \bar{u}_k \\ & x_0 \in \bar{x}_0 \oplus \mathcal{Z} \\ & \bar{u} \in \mathcal{U} \ominus K \mathcal{Z} \\ & \bar{x} \in \mathcal{X} \ominus \mathcal{Z} \end{aligned} \quad (2)$$

where Q , R and Q_N are the state, control and terminal state cost matrices, respectively. The control command given would be $u = \bar{u} + K(x - \bar{x})$, where x is the current state. This guarantees $x^+ \in \bar{x}^+ \oplus \mathcal{Z}$ for any $w \in \mathcal{W}$, i.e., all states x_k will be inside the constraint set \mathcal{X} . However, for a nonlinear system as in our case, we use the equivalent LTV system, where the system matrices A and B are replaced with Jacobian matrices \hat{A} and \hat{B} at the current state for the system dynamics used in Equation 1. For more details as well as a detailed proof of the feasibility and the stability of the above controller, see [11]. Also, mismatch between the linearized and the actual model can be compensated by adding an additional disturbance \hat{w} assuming the model non-linear function to be a Lipschitz function [12]. For the experiments in this work, we assume the disturbance margin is large enough to cover this extra disturbance.

D. Delay-aware robust tube MPC

For the above formulation, we assumed delay time to be zero, meaning that the computed command is delivered to

the system at the same time the observation is made for the current state x . But in practice, there is computation time denoted as t_c and an actuator command processing delay t_a after the calculated command is delivered, resulting in a total delay time $t_d = t_c + t_a$. Hence, if the current state x is observed at time t , the computed command influences the actuator state at $t + t_d$ time. This may lead to instability of the system if t_d is large and the robust tube assumptions no longer hold true. In order to tackle this problem, Ref. [7] proposes a bi-level control scheme to deal with time delay and also proves robustness using the tube MPC. The high-level MPC runs to get the robust tube for the next cycle with the initial tube position constraint to cover all possible initial state positions under bounded disturbances. Meanwhile, the second-level control scheme acts as a pre-compensator unit and ensures robustness by correcting the current state towards the nominal state calculated from the first level. A buffer block of commands is used for communication between higher- and lower-level units, as depicted in Figure 1. At time t , the set of possible states at time $t + t_h$, where t_h is the horizon length, is predicted. The high-level tube MPC updates the buffer with nominal states and control commands from time $t + t_h$ to $t + 2t_h$. If the higher-level MPC requires a delay time t_c that is less than t_h , the system waits for the remaining time $t_h - t_c$. However, we believe this is only suited for systems when the objective is nearly constant. For dynamically changing objectives as well as state constraints, it is also necessary to update the reference path more frequently, since waiting for the full horizon path to be followed may lead to inconsistencies. In the case of autonomous driving, for dynamic scenarios where the reference path has to be updated frequently, it would not be feasible to wait for t_h to get a new updated path.

We propose to get a local probabilistic upper-bound estimate \hat{t}_c of the computation time. We update the buffer from $(t + \hat{t}_c)$ to $t + \hat{t}_c + t_h$ instead of from $(t + t_h)$ to $t + 2t_h$ as shown in Fig. 1. This increases the controller plan update rate for the higher-level MPC and also makes the controller robust to changing computation times. For estimation of the local upper bound \hat{t}_c , we use a Kalman filter as further described in Section III-E. Considering t_a to be the extra delay due to actuator command processing, thus getting a new local upper-bound delay time estimate $\hat{t}_d = \hat{t}_c + t_a$, which we use to find the initial state estimate $x_{\hat{t}_d|t}$ assuming no disturbance after \hat{t}_d time given the current state x_t . It can be calculated by piece-wise integration of the system dynamics using the control commands from the buffer. Hence, command executed at $t + \hat{t}_c$ will be executed at $t + \hat{t}_c + t_a = t + \hat{t}_d$ where we consider our initial state to be for optimization as depicted in Fig. 1. Mathematically, the updated objective function is described in Equation 3, where $x_{\hat{t}_d+t}$ is the actual state after time \hat{t}_d . The calculated nominal discrete states (\bar{X}) and controls (\bar{U}) are used to fill the buffer B from time $t + \hat{t}_c$ to $t + \hat{t}_c + t_h$ as $\bar{u}_{[t+\hat{t}_c+k\Delta t, t+\hat{t}_c+(k+1)\Delta t]} = \bar{u}_k$ for $k \in \{0, 1, 2, \dots, N-1\}$, as shown in Figure 1. Pre-compensator unit is a low level process which executes command $c = \bar{u}_t' + K(x_t' - \hat{x}_t')$ at time t' in the buffer at high frequency.

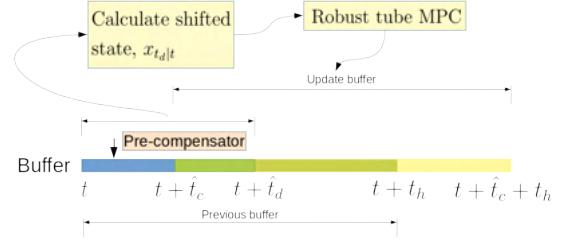


Fig. 1: Dual cycle control scheme for tube MPC with delay

$$\begin{aligned}
 \min_{\bar{x}_0, \bar{U}} \quad & \sum_{t=0}^{N-1} (\bar{x}_k - x_{ref,k})^T Q (\bar{x}_k - x_{ref,k}) + \bar{u}_k^T R \bar{u}_k \\
 & + (\bar{x}_N - x_{ref,N})^T Q_N (\bar{x}_N - x_{ref,N}) \\
 \text{s.t.} \quad & \bar{x}_{k+1} = A \bar{x}_k + B \bar{u}_k \\
 & x_{t_d+t} \in \bar{x}_0 \oplus \mathcal{Z} \implies x_{t_d|t} \in \bar{x}_0 \oplus \mathcal{Z} \ominus (\bigoplus_{j=0}^{s-1} A_k^j \mathcal{W}) \\
 & s = \left\lceil \frac{\hat{t}_d}{\Delta t} \right\rceil \\
 & \bar{u} \in \mathcal{U} \ominus K \mathcal{Z} \\
 & \bar{x} \in \mathcal{X} \ominus \mathcal{Z}
 \end{aligned} \tag{3}$$

1) *Control Constraints (\mathcal{U}):* Limits on acceleration and steering are formulated as control constraints for the optimization problem.

2) *State Constraints (\mathcal{X}):* For the state variables θ and v , we set the upper and lower bound for their range. However, for the state variables p_x and p_y , free space is non-convex in nature, hence it becomes quite computationally expensive to set them in the non-convex form for the optimization problem. Hence we use the IRIS algorithm we have used in previous work [13] to derive a set of convex constraints which can be used for efficient optimization of the path tracking problem while also ensuring safety through collision avoidance. IRIS optimizes the objective of finding linear constraints for each obstacle such that the resultant convex space fits the largest possible ellipsoid. We set the seed for IRIS as the predicted position (without uncertainty) of the vehicle after time \hat{t}_d to get the resultant convex space \mathcal{X} .

3) *Disturbance-invariant set (\mathcal{Z}):* The disturbance-invariant set can be over-approximated using $\mathcal{Z}_k = \bigcup_{i=0}^N A_k^i \mathcal{W}$ for a given A_k [14]. However, in the presence of delay, A_{k+1} would be different for the next control cycle, hence for robustness, \mathcal{Z}_k must be sufficient to be covered by \mathcal{Z}_{k+1} .

Theorem 1. *Given the optimization in Equation 3, the disturbance-invariant set \mathcal{Z}_k is calculated as $\mathcal{Z} = \bigcup_{A \sim S_A} (\bigoplus_{j=0}^{N-1} A^j \mathcal{W})$, i.e., a union set with all possible values of initial heading angle, speed and steering angle within the admissible range, which determines the Jacobian matrix A_k . The invariant set guarantees robust initialization of the optimization problem.*

Proof. $x_{t_d|t_k}$ is the expected state assuming no disturbance after t_d for the current time t_k , and the application of \bar{u} with feedback. $x_{t_d+t_k}$ is the actual state at t_d+t_k . We have $x_{t_d+t_k} \in$

$x_{t_d|t_k} \oplus (\bigoplus_{j=0}^{s-1} A_k^j \mathcal{W})$, where $s = \lceil \frac{t_d}{\Delta t} \rceil$. From Equation 3, we have $x_{t_d+t_k} \in x_{0|t_d+t_k} \oplus \mathcal{Z}$. Hence, in order to guarantee robustness, $x_{t_d|t_k} \in \bar{x}_{0|t_d+t_k} \oplus \mathcal{Z} \ominus (\bigoplus_{j=0}^{s-1} A_k^j \mathcal{W})$, we need to ensure $\bar{x}_{0|t_d+t_k}$ have a valid solution, i.e., $(\bigoplus_{j=0}^{s-1} A_k^j \mathcal{W}) \subseteq \mathcal{Z}$. As $s \leq N$ and $0_n \in \mathcal{W}$, we can establish

$$(\bigoplus_{j=0}^{s-1} A_k^j \mathcal{W}) \subseteq (\bigoplus_{j=0}^{s-1} A_k^j \mathcal{W}) \oplus (\bigoplus_{j=s}^{N-1} A_k^j \mathcal{W}) \quad (4)$$

Thus, $\bigoplus_{j=0}^{s-1} A_k^j \mathcal{W} \subseteq \bigoplus_{j=0}^{N-1} A_k^j \mathcal{W}$. A_k is the Jacobian matrix which depends on θ_k , v_k , and δ_k of $x_{t_d|t_k}$. Let's define set $S_A = \{A(\theta, v, \delta) | \theta \in [-\pi, +\pi], \delta \in [-\delta_{max}, \delta_{max}], v \in [v_{min}, v_{max}]\}$, which consists of all possible matrices.

$$(\bigoplus_{j=0}^{N-1} A_k^j \mathcal{W}) \subseteq \bigcup_{A \sim S_A} (\bigoplus_{j=0}^{N-1} A^j \mathcal{W}) \quad (5)$$

Thus, $\bigoplus_{j=0}^{s-1} A_k^j \mathcal{W} \subseteq \bigcup_{A \sim S_A} (\bigoplus_{j=0}^{N-1} A^j \mathcal{W})$. \mathcal{Z} is chosen as $\bigcup_{A \sim S_A} (\bigoplus_{j=0}^{N-1} A^j \mathcal{W})$, which concludes the proof. \square

E. Estimating computation time

For estimating a local upper bound on computation time, we use an Adaptive Kalman Filter (AKF) on the single variable. We initialize the estimated computation time and the variance associated with it by the first measurement. For the Kalman filter, we need state update parameters γ , process noise variance q and measurement noise variance r . We assume both noise distributions to be gaussian, independent and mutually uncorrelated throughout. The adaptive kalman filter, uses data points to estimate their values recursively at each step. Running average of prediction error e and measurement error w are used to recursively update the values of q and r . Parameters N_p , N_q , N_θ determine roughly the no of previous samples effectively taken for the exponential weighted mean of e and w which are used for update of q and r respectively. These update rules have been adapted from [15]. While the update rules for state transition parameters θ have been adapted from [15]. $\lambda = \frac{N_q-1}{N_\theta}$ denotes the forgetting factor for estimation of θ . The following steps are performed in a loop (see Equation 6). $(.)_{n|n-1}$ and $(.)_{n|n}$ denotes the prior and posterior estimates at n^{th} step.

$$x_{n|n-1} = \gamma_{n-1} [x_{n-1|n-1} \ 1]^T \quad (6a)$$

$$p_{n|n-1} = p_{n-1|n-1} + q_{n-1} \quad (6b)$$

$$e_n = \frac{N_r - 1}{N_r} e_{n-1} + \frac{1}{N_r} (z_n - x_{n|n-1}) \quad (6c)$$

$$\Delta r_n = \frac{((z_n - x_{n|n-1}) - e_n)^2}{N_r - 1} - \frac{p_{n|n-1}}{N_r} \quad (6d)$$

$$r_n = \left| \frac{N_r - 1}{N_r} r_{n-1} + \Delta r_n \right| \quad (6e)$$

$$K_n = \frac{p_{n|n-1}}{p_{n|n-1} + r_n} \quad (6f)$$

$$x_{n|n} = x_{n|n-1} + K_n (z_n - x_{n|n-1}) \quad (6g)$$

$$p_{n|n} = (1 - K_n) p_{n|n-1} \quad (6h)$$

$$w_n = \frac{N_q - 1}{N_q} w_{n|n-1} + \frac{1}{N_q - 1} (x_{n|n} - x_{n|n-1}) \quad (6i)$$

$$\Delta q_n = \frac{p_{n|n} - p_{n-1|n-1}}{N_q} + \frac{(x_{n|n} - x_{n|n-1}) - w_n}{N_q - 1} \quad (6j)$$

$$q_n = \left| \frac{N_q - 1}{N_q} q_{n-1} + \Delta q_n \right| \quad (6k)$$

Let ϕ be $[x_{n-1|n-1} \ 1]^T$

$$F_n = \frac{1}{\lambda} \left(F_{n-1} - \frac{F_{n-1} \phi \phi^T F_{n-1}}{\lambda + \phi^T F_{n-1} \phi} \right) \quad (6l)$$

$$\gamma_n = \gamma_{n-1} + F_n \phi (x_{n|n} - x_{n|n-1}) \quad (6m)$$

For the local upper bound estimate, we use the predicted value and variance to get an upper-bound estimate on computation time at step k , $t_{c,k}$ (Equation 7). We choose the parameter β accordingly to get sufficiently high confidence as an upper bound assuming a Gaussian distribution.

$$\hat{t}_{c,n} = x_{n|n} + \beta p_{n|n} \quad (7)$$

F. Controller Plan A

We present a standalone controller (called plan A) as depicted in Figure 2. We compensate for the actuator dynamics by including actuator first-order ODE as part of the system dynamics. For the compensation of computation and actuator command processing delays, we use initial state shift by the estimated local upper bound on the net delay time. The optimization problem updates the robust tube buffer from $t + \hat{t}_d$ to $t + \hat{t}_d + t_h$ with the nominal commands and states. The pre-compensator unit runs as a low-level process to refine the control to maintain the robust tube condition, see Figure 2.

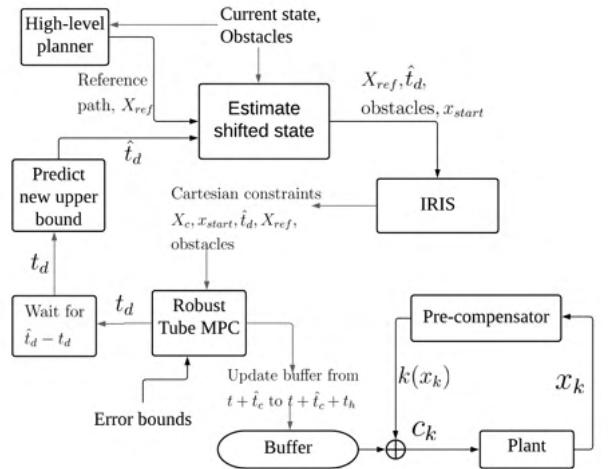


Fig. 2: Controller Plan A for Robust tube MPC

G. Controller Plan B

As an alternate plan, we compensate for the actuator and computation delay of a nominal controller. It can be a black-box controller, which doesn't consider compensation for these delays. This plan can be used for a LE controller which has been trained on the simplified simulation environment without all practical delays. We use a separate unit which takes the commands from the nominal controller as reference to track. Assuming the commands from the LE controller

be $\hat{U} = \{\hat{u}_1, \hat{u}_2, \dots, \hat{u}_N\}$, the output is the set of refined commands after compensating for actuator dynamic delay as $U = \{u_1, u_2, \dots, u_N\}$. U is obtained by solving a quadratic optimization problem (Equation 8) where u_{start} is the current value of the steering angle, r_k is the unit step response of the steering actuator at the k^{th} time step, and Q_{ac} and R_{ac} are positive semidefinite weight matrices. The optimization is to track the desired actuator commands from the LE controller as closely as possible while minimizing the control effort. For the computation time compensation, we use the same design as plan A by shifting the initial state, see Figure 3.

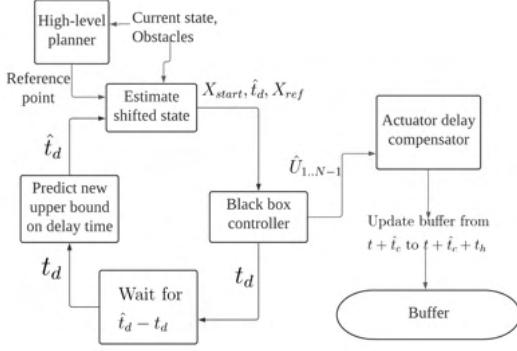


Fig. 3: Controller Plan B for black box controller

$$\begin{aligned} \min_U & \sum_{k=1}^N \left\| \hat{u}_k - \left(u_0 + \sum_{l=1}^{k-1} (u_l - u_{l-1}) r_{k-l+1} \right) \right\|_{Q_{ac}} \\ & + \|u_k\|_{R_{ac}} \\ \text{s.t. } & u_0 - u_{start} = 0 \\ \text{where } & r_l = (1 - e^{-Kl\Delta t}). \end{aligned} \quad (8)$$

IV. EXPERIMENTAL RESULTS

We conduct the experiments in the Gazebo simulator with a Prius vehicle model. In order to get the time constant value for the steering actuator, we test the unit response of the actuator, i.e., we set the actuator command to 1 and record the steering angle values over a time window sufficient for the steering angle to converge at the maximum possible value. We then fit the observed response values with the first-order ODE described in Section III-B and determine the parameter K . As shown in Figure 4, using $K = 30$ well approximates the actuator dynamics for the Prius model in Gazebo.

A. Static scenario

For testing controller plan A, we perform a static obstacle avoidance experiment. The planner used is hybrid A*, [16]. We compare the paths followed by the controller with and without considering delay time and actuator dynamics. We also compare the paths followed if the delay time is taken as an upper bound equal to the horizon length, which is proposed in ?? and if the local upper bound on the delay time is approximated using our proposed Kalman filter. The path followed using our proposed method can be clearly seen to be better. This is because at point A (Figure 5), the state

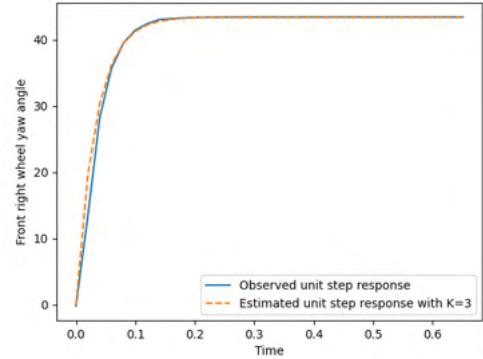


Fig. 4: Steering angle response for Prius model in gazebo

constraints generated from IRIS as shown in the Figures (5 6 7) force the vehicle to deviate from the reference path, which thus overshoots by a significant amount, but is still able to get back to the reference, which is not the case if we don't consider compensation for any time delay. On the other hand, if we approximate the delay time using a Kalman filter and adjust the expected local upper bound value accordingly (see the delay estimation in Figure 6), the controller responds faster. Hence, after passing point A, the state constraints change and the controller reacts faster to get back to the reference path, giving less overshoot. The robust tube condition gets violated for the controller if the compensation for actuator dynamics and delay time is not considered at point B, when the vehicle collides with the obstacle.

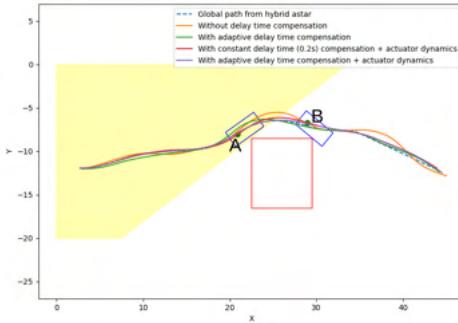


Fig. 5: Comparison on paths followed with and without delay compensation. Yellow region is the convex state constraint set from IRIS at point A of the trajectory for Experiment IV-A.

B. Overtaking scenario

We further test the controller plan A in an overtaking scenario. For this scenario, the lead vehicle brakes suddenly at point A, $t = 4s$. For this scenario, we use the Frenet planner [17] with the reference path as the lane center. Frenet frame-based planning has been successful in practice due to the significant advantage of its independence from complex road geometry. We perform the experiment with the same starting conditions and compare the results with (Figure 8a) and without (Figure 8b) delay compensation. The Frenet path

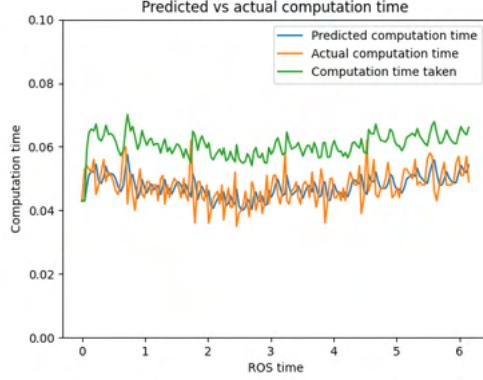


Fig. 6: Observed computation times and their estimated values using Adaptive Kalman filter (AKF) for Experiment IV-A.

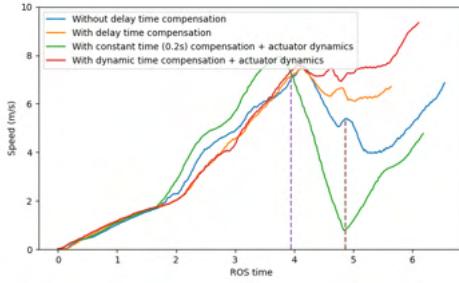
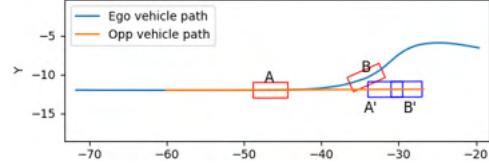


Fig. 7: Speed comparison for the paths followed with and without delay compensation for Experiment IV-A.

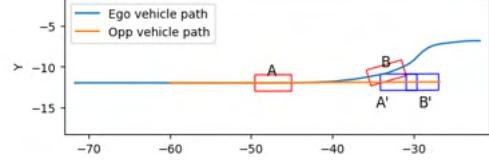
planner expects the ego vehicle to move at constant speed, but as the speed rapidly drops at point A, as shown in the figures, the reference path changes rapidly. The Frenet planner thus rapidly changes path after point A. Point B is the closest position between the ego vehicle and lead vehicle in all the cases. If delay time is not considered, the ego vehicle hits the other vehicle slightly at point B. Also, in this case if computation time is taken as a constant upper bound of 0.2s, due to the slow reaction of the controller, the ego vehicle hits the lead vehicle at point B (Figure 8b). Hence, it can be seen that taking the computation time as an upper bound might be ineffective in rapidly changing scenarios.

C. Closed track scenario

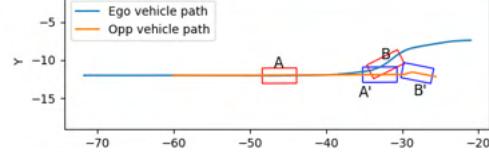
Finally, we test our controller plan B on a LE controller which has been trained on an ideal environment without delay. The LE lateral controller is a neural network trained on waypoint following with inputs $[\Delta x, \Delta y, \Delta \theta]$, where $(\Delta x, \Delta y)$ and $\Delta \theta$ are respectively the vector co-ordinates and direction of the target waypoint in the local frame of reference of the ego vehicle. The output is the steering angle δ . The network architecture is a simple feed-forward neural network with hidden layer sizes (4, 16, 4). For longitudinal control, simple PID control is used to track a constant speed of 75 m/s throughout. When deploying the controller in Gazebo for waypoint-following, the controller performs worse due to the practical delays. The controller deployed without compensation loses control at the time of turn, as demonstrated in



(a) With variable local upper bound on computation delay.



(b) Without computation delay consideration.



(c) With constant upper bound on computation delay (0.2s).

Fig. 8: Comparison in paths traced by the ego vehicle and the traffic vehicle for Experiment IV-B

Figure 9. After using the proposed plan B, which contains compensation for the computation and actuator delays, the vehicle retains control. The vehicle is operating at its friction limits, hence even a little bit of error caused due to delay leads to vehicle losing control even when the computation time is just 0.02s on an average during the experiment.

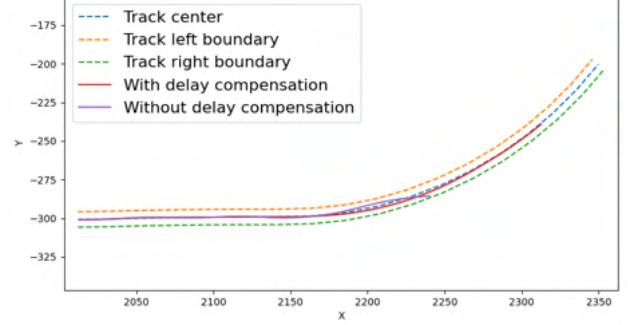


Fig. 9: Comparison in paths followed with and without delay compensation on a closed track centre line tracking objective for Experiment IV-C

V. CONCLUSION

In this work, we propose a unified framework for compensating the delays of computation, actuator command processing and actuator dynamics in autonomous driving systems. We use a Kalman filter for approximating the computation time instead of taking it as an upper bound, leading to better performance and response in dynamic scenarios. With the use of tube MPC, the vehicle is able to safely track the planned trajectories in realistic scenarios tested in the high-fidelity Gazebo simulator. Lastly, we also present a general framework for compensating delays in real environments for

any black-box controller which works in an ideal environment. The control framework has been tested on a system with a 1.60GHz Intel Core i5-8250U CPU. The simulation results demonstrate safety and real-time performance of our proposed framework.

REFERENCES

- [1] Z. M. U. Din, W. Razzaq, U. Arif, W. Ahmad, and W. Muhammad, “Real time ackerman steering angle control for self-driving car autonomous navigation,” in *2019 4th International Conference on Emerging Trends in Engineering, Sciences and Technology (ICEEST)*. IEEE, 2019.
- [2] R. C. Conlter, “Implementation of the pure pursuit path’hecking algorithm,” 1992.
- [3] M. Rokonuzzaman, N. Mohajer, S. Nahavandi, and S. Mohamed, “Review and performance evaluation of path tracking controllers of autonomous vehicles,” *IET Intelligent Transport Systems*, vol. 15, no. 5, pp. 646–670, 2021.
- [4] F. Borrelli, P. Falcone, T. Keviczky, J. Asgari, and D. Hrovat, “Mpc-based approach to active steering for autonomous vehicle systems,” *International journal of vehicle autonomous systems*, vol. 3, no. 2-4, pp. 265–291, 2005.
- [5] Y. Liao and F. Liao, “Design of preview controller for linear continuous-time systems with input delay,” *International Journal of Control, Automation and Systems*, vol. 16, no. 3, pp. 1080–1090, 2018.
- [6] P. Cortes, J. Rodriguez, C. Silva, and A. Flores, “Delay compensation in model predictive current control of a three-phase inverter,” *IEEE Transactions on Industrial Electronics*, vol. 59, pp. 1323–1325, 2011.
- [7] Y. Su, K. K. Tan, and T. H. Lee, “Computation delay compensation for real time implementation of robust model predictive control,” *Journal of Process Control*, vol. 23, no. 9, pp. 1342–1349, 2013.
- [8] A. Nahidi, A. Khajepour, A. Kasaeizadeh, S.-K. Chen, and B. Litkouhi, “A study on actuator delay compensation using predictive control technique with experimental verification,” *Mechatronics*, vol. 57, pp. 140–149, 2019.
- [9] N. E. Kahveci and P. A. Ioannou, “Automatic steering of vehicles subject to actuator saturation and delay,” in *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2011, pp. 119–124.
- [10] A *Multi-Stage Optimization Formulation for MPC-Based Obstacle Avoidance in Autonomous Vehicles Using a LIDAR Sensor*, ser. Dynamic Systems and Control Conference, vol. Volume 2: Ground and Space Vehicle Dynamics, 10 2014.
- [11] R. Smith, “Robust model predictive control of constrained linear systems,” in *Proceedings of the 2004 American Control Conference*, vol. 1, 2004, pp. 245–250 vol.1.
- [12] Y. Gao, A. Gray, H. E. Tseng, and F. Borrelli, “A tube-based robust nonlinear predictive control approach to semiautonomous ground vehicles,” *Vehicle System Dynamics*, vol. 52, no. 6, pp. 802–823, 2014.
- [13] S. Khatan, Q. Lin, and J. M. Dolan, “Safe planning and control under uncertainty for self-driving,” *arXiv preprint arXiv:2010.11063*, 2020.
- [14] S. V. Raković, E. Kerrigan, K. Kouramas, and D. Mayne, “Invariant approximations of robustly positively invariant sets for constrained linear discrete-time systems subject to bounded disturbances,” 02 2004.
- [15] I. Hashlamon and K. Erbatur, “An improved real-time adaptive kalman filter with recursive noise covariance updating rules,” *Turkish Journal of Electrical Engineering and Computer Sciences*, 12 2013.
- [16] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, “Practical search techniques in path planning for autonomous driving,” *Ann Arbor*, vol. 1001, no. 48105, pp. 18–80, 2008.
- [17] M. Werling, J. Ziegler, S. Kammler, and S. Thrun, “Optimal trajectory generation for dynamic street scenarios in a frenet frame,” in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010.

AirObject: An Evolving Topological Graph-based Object Encoding for Semantic Loop Closure

Nikhil Varma Keetha^{*,1}, Chen Wang², Yuheng Qiu² and Sebastian Scherer²

Abstract—Object encoding and identification in a distributed setup are vital for multi-agent robotics tasks involving autonomous exploration, semantic scene understanding, and relocalization. Previous approaches have attempted to either track object detections or generate embeddings of an object from a single viewpoint for object identification. However, such systems are limited to a “fixed” object representation from a single viewpoint and agent. Furthermore, in an online distributed setup such as Multi-agent Semantic Simultaneous localization and mapping (SLAM), there is a requirement for an “evolving” object representation across multiple agents and viewpoints that is shared and merged effectively. In this paper, we propose a novel incremental 3D object encoding approach to obtain global graph-based keypoint representations and embeddings of objects. We employ topological graph merging and a hierarchical graph representation-based encoding method to generate global object descriptors robust to drastic viewpoint shift, occlusion, deformation, and scale of the representation, either across multiple agents and viewpoints or from a single agent and viewpoint. We demonstrate that our approach achieves great performance for object identification and semantic loop closure.

Index Terms—Localization; Semantic Scene Understanding; Deep Learning Methods; Recognition

I. INTRODUCTION

Object encoding and identification are crucial for many robotics tasks such as autonomous exploration, semantic scene understanding, and loop closure in simultaneous localization and mapping (SLAM). For example, drift-free large-scale object-based semantic SLAM and identification of revisited interesting landmark objects require robust and efficient object encodings [1], [2]. Prior approaches proposed in the literature have attempted to track object detections [3], use keypoint features [4] or generate graph-based embeddings of an object from a single viewpoint for object matching [5]. However, such systems are limited to a “fixed” object representation from a single viewpoint and robotic agent and are not robust to drastic viewpoint shift or scale of representation. Furthermore, in a Multi-agent Semantic SLAM setup [6], existing methods easily produce false matches and incorrect inter-robot loop closures. Hence, in this context, an object encoding method to generate robust object representations that can be shared and merged effectively across multiple viewpoints and agents is necessary.

In this work, we propose a novel topological graph merging-based 3D object encoding method to generate tem-

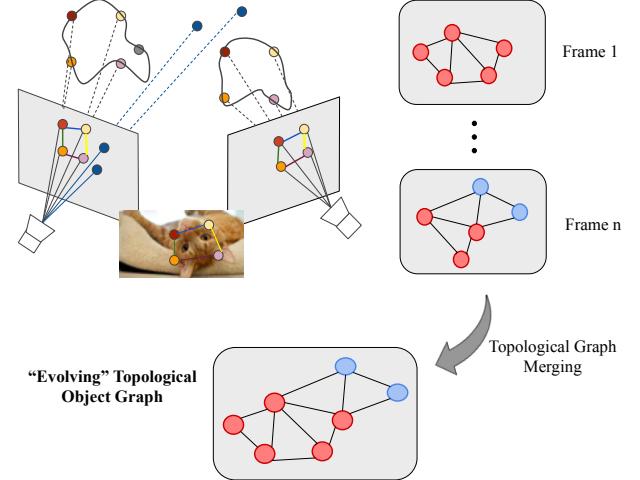


Fig. 1. We propose a topological graph merging method that builds a temporally evolving object graph where new nodes are added as different viewpoints or distinct parts of an object are explored.

porally evolving representations of objects (see Fig 1). Specifically, we use extracted deep learned keypoint features [7] to form object-wise topological graph neural networks (GNNs), which on embedding generate single-viewpoint object descriptors. To deal with a varying number of object keypoints, we employ a sparse object encoding method [5] so that only a few positions of an object descriptor can be affected due to change in keypoints. Furthermore, the topological GNN and sparse object encoder generate node-wise location descriptors and feature descriptors. These node-wise location descriptors are used for topological graph merging across multiple viewpoints of an object to build incrementally growing topological object graphs. These temporally “evolving” topological object graphs are embedded to generate 3D object descriptors which accumulate knowledge across multiple viewpoints making the descriptors robust to viewpoint changes, the scale of representation, and deformation.

In summary, we make the following specific contributions:

- Delaunay triangulation based topological object graph representations which leverage explicit geometry from keypoints;
- a topological graph merging method based on sparse location descriptors to build a temporally “evolving” topological object graph; and
- a method to generate 3D object descriptors that accumu-

¹The author is with the Indian Institute of Technology (ISM) Dhanbad, India. email: keethanikhil@gmail.com

²The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. email: {chenwan3, yuhengq, basti}@andrew.cmu.edu

*This work was done as a CMU Robotics Institute Summer Scholar.

late knowledge across multiple viewpoints and provide reliable object matching and semantic loop closure.

II. RELATED WORK

In this section, we review object matching, loop closure methods based on handcrafted and deep learned features. Furthermore, we also review Visual Place Recognition (VPR) methods that can be extended to object identification. Object tracking methods based on networks such as Mask R-CNN [8] are not included because they are not suitable for object re-identification.

A. Handcrafted Features

Handcrafted features such as SIFT [9] and SURF [10] have been widely used in classical approaches for loop closure, object matching, and VPR. One such classical approach, fast appearance-based mapping (FABMAP) [11], utilizes the trained visual vocabulary of SURF features, obtained by hierarchical k-means clustering, for identifying revisited objects through feature distribution matching. Further extending this idea, a binary descriptor ORB [12] was used in DBoW2 [13] to achieve better speed. Furthermore, in recent work such as Kimera-VIO [14] and Kimera-Multi [6], a Bag-of-Words (BoW) representation of keyframes based on ORB features and DBoW2 was used for distributed loop closure detection.

Further building on the idea of vocabulary-based retrieval, [15] introduces a vocabulary maintenance strategy that groups similar images by repetitive registration of the same descriptors across multiple frames. To further speed up the retrieval process, [16] introduces an online trained Hamming distance embedded binary search tree (HBST). However, the memory cost of the incremental visual vocabulary tree is huge due to the use of raw local descriptors. This memory overhead can be mitigated through dimensional reduction. One such approach, [17], employs principle component analysis (PCA) on BRISK [18] features and applies k-nearest neighbor (K-NN) search on the projected descriptors to achieve fast speeds in the range of milliseconds for each query while maintaining a low memory overhead. However, these handcrafted features based methods are sensitive to environmental changes and lead to false matches when the local descriptors are not discriminative enough.

B. Deep learned Features

The recent success of convolutional neural networks (CNN) [19] in computer vision has led to the rise of deep learned features-based image retrieval. The methods employing deep learned features have shown tremendous improvements over handcrafted features. One such method, [20], uses a multi-scale feature encoding across two CNN architectures to generate CNN features that are viewpoint invariant, thereby providing large performance improvement. Another popular end-to-end deep learning-based approach, NetVLAD [21] generates descriptors inspired by the traditional vector of locally aggregated descriptors (VLAD). Further building on both feature-based and deep learning

methods, [22] uses a Multi-Process Fusion to combine different image processing methods such as the sum of absolute differences, histogram of oriented gradients [23], CNN spatial Max pooling, and CNN spatial Arg-Max Pooling for VPR.

Further exploring other input modalities such as RGB-D images, [24] incorporates depth information of objects into pre-trained CNN features by rendering objects from a canonical perspective and colorizing the depth channel based on distance from object center. Similarly, [25] embeds point cloud and depth data into the RGB domain for RGB-D object recognition. Another recent RGB-D-based method, HP-CNN [26], uses multi-view 3D object poses from RGB-D sensors to generate multi-scale object feature representations.

Recently proposed deep learning method, SuperPoint [7] leverages a self-supervised framework to train interest point detectors and descriptor extractors. Further building on SuperPoint, SuperGlue [27] introduced a local feature matcher based on graph attention [28] where the interest points are nodes of a graph, and their associated descriptors are the node features. Both SuperPoint and SuperGlue have been widely adopted for the task of feature matching and hierarchical VPR [29], [30]. Similar to SuperGlue, [5] embeds object-wise fully connected graph-based representations of SuperPoint features using a Sparse Object Encoder to generate object descriptors. However, this approach doesn't consider the explicit geometry available from the SuperPoint interest points when constructing the graph-based representations of objects. Also, it generates fixed object representations limited to only a single viewpoint. In this context, our proposed framework builds temporally evolving topological graph representations of objects to generate 3D object descriptors that accumulate knowledge across multiple viewpoints.

III. PROPOSED APPROACH

In this section, we first present the proposed method to generate topological graph representations of objects that are input to the object encoder. We then describe the structure of the object encoder and our approach to building and embedding evolving topological graph representations of objects to generate 3D object descriptors (see Fig 2).

A. Topological Graph-based Object Representations

Intuitively, a group of feature points for an object form a graphical representation where the feature points are nodes and their associated descriptors are the node features. Essentially, the graph's nodes are the local distinctive features of the object, while the edges/structure of the graph represents the global structure of the object. We believe that embedding such a topological graph-based representation of an object containing both local distinctive features and global object structure will enable robust object identification similar to humans [31]. Hence based on this hypothesis, we formulate a procedure to generate topological graph representations of feature points corresponding to objects.

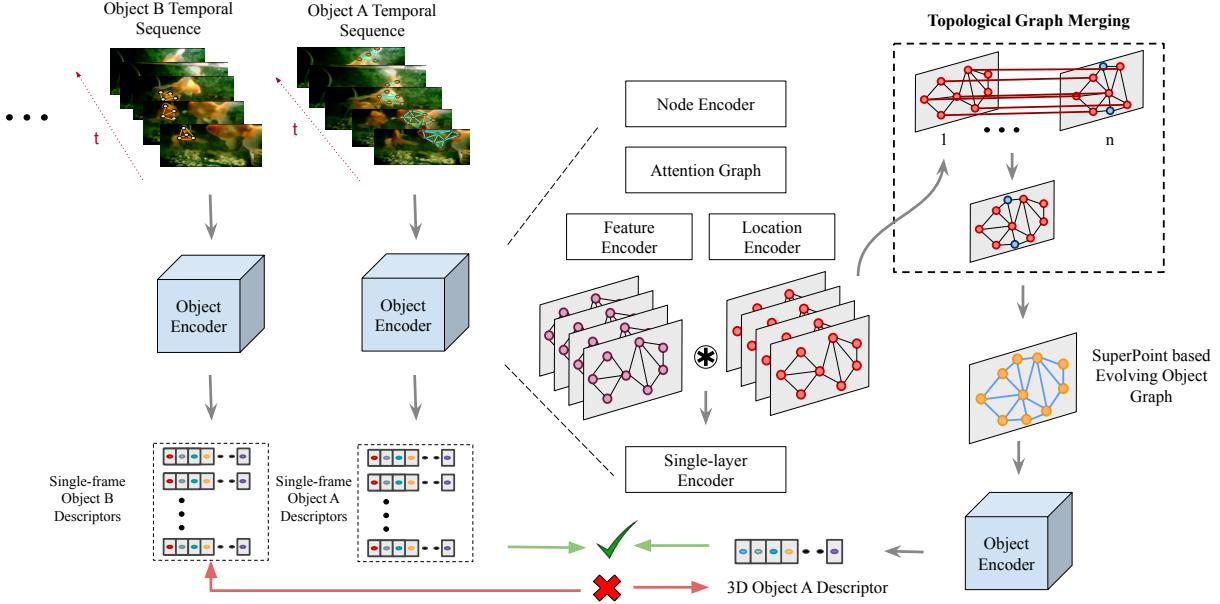


Fig. 2. Schematic of our proposed approach. The SuperPoint based topological graph representations of objects are input to the object encoder to generate the node-wise location features and the single-frame object descriptors. Then, based on these node-wise location features, topological graph merging is performed to create a SuperPoint based evolving graph representation of the object. This evolving topological graph is input into the object encoder to generate the 3D object descriptor, which provides positive matches with the descriptors of the same object and rejects false matches.

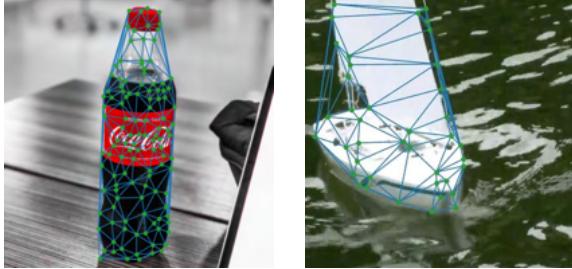


Fig. 3. Topological Graph Representations of Objects. These representations are generated by using Delaunay Triangulation on object-wise grouped SuperPoint keypoints.

Given an object, we extract a set of feature points corresponding to the object, where the position of each feature point is denoted as $p_i = (x, y), i \in [1, M]$ and the associated descriptor as $d_i \in \mathbb{R}^{N_p}$, where N_p is the descriptor dimension. In practice, these object-wise grouped feature points are obtained by using ground-truth instance segmentation masks or object segmentation masks from pretrained Mask R-CNN [8] along with point detector SuperPoint [7]. Given these object-wise grouped feature points, our goal is to generate a topological graph representation that leverages the explicit geometry provided by the position of each feature point.

Essentially, we build a topological graph structure for the feature points corresponding to an object by using Delaunay triangulation [32] on the positions of the feature points, as shown in Fig 3. Delaunay triangulation is a mathematical formulation, where given a set of discrete points, the objective is to provide a triangulation such that no discrete

point is present inside the circumcircle of any triangle of the triangulation. This objective can be particularly stated as maximizing the minimum angle of all the angles of the triangles in the triangulation, thereby avoiding narrow and intersecting triangles. This particular property coupled with fast processing speeds makes Delaunay triangulation desirable to generate mesh representations that model the object's geometry. We believe that inputting a topological graph representation of object feature points based on this mesh structure to the object encoder will enable the object encoder to reason better about the global structure/geometry of the object and further make encoding of temporally evolving topological graph representation of the object robust to deformation or occlusion.

B. Object Encoder

The topological graph representation of the object feature points is input to the GNN based sparse object encoder [5]. A particular desirable property of a sparse object encoder is that a single keypoint should only have a local effect on sparse locations of the object descriptor so that addition or removal of keypoints doesn't significantly change the object descriptor. Furthermore, the object encoder should encode distinct keypoints to unique locations within the object descriptor. Essentially, to facilitate this, the object encoder is comprised of a Node Encoder, a two-layer Graph Attention (GAT) [28] module followed by a Sparsity module which contains two parallel heads, namely a feature encoder and a location encoder whose outputs are element-wise multiplied and passed through a single-layer encoder to generate object descriptors.

The Node Encoder takes the positions and descriptors of the topological graph-based object representation and encodes the graph's node features x_i as the concatenation of the point descriptor and transformation of the point position:

$$x_i^{(1)} = [d_i \parallel MLP(p_i)], \quad x_i \in \mathbb{R}^{N_n}, \quad (1)$$

where $N_n = N_p + N_m$, \parallel denotes the concatenation operator, the Multi-layer Perceptron (MLP) module maps $\mathbb{R}^2 \mapsto \mathbb{R}^{N_m}$, and superscript of x_i^l denotes the l -th layer of the GNN. In practice, we normalize the position p_i to $[-1, 1]$ by the object size, which considers the object center as the origin. The concatenation of the transformation of the position as opposed to a summation similar to SuperGlue [27] helps the object encoder explicitly learn the object structure since the relative position information is not blended into the descriptor. Furthermore, this concatenation operation enables the Sparsity module of the object encoder to learn sparse non-zero locations for the keypoints based on the object structure.

The graph's node features x_i and the adjacency matrix from the topological graph structure are input to a two-layered GAT to enable structured attention-based message propagation between the features of the object¹. This helps the object encoder reason about global feature interactions between the local distinctive keypoint-based features of the object. Afterward, the output of the GAT is passed on to the Sparsity module to encode the graph embeddings as an object descriptor such that learned location features of the node decide the sparse location of the keypoint on the object descriptor.

The Sparsity module of the object encoder contains two parallel heads where each head contains two stacked sparsity layers to learn the location feature x_i^L and the content feature x_i^C , whose inputs are x_i from the GAT (here we leave out the layer index (l) for simplicity). The sparsity layers for the location node features and content node features are defined as:

$$(l+1)x_i^L = \text{ReLU}(W_L^{(l)} \cdot {}^{(l)}x_i^L), \quad (2a)$$

$$(l+1)x_i^C = \text{ReLU}(W_C^{(l)} \cdot {}^{(l)}x_i^C), \quad (2b)$$

where $W_L^{(l)}$, $W_C^{(l)} \in \mathbb{R}^{N_o \times N_n^l}$, $N_n^l < N_o$ are the learnable location and content weights, respectively, and N_o is the dimension of the object descriptor. Then, these node-wise location and content features are used to generate the object descriptor D_k in the following way:

$$D_k = W_o \sum_{i=1}^M x_i^L \odot x_i^C, \quad (3)$$

where W_o is a learnable single-layer encoder, M is the number of nodes in the graph, and \odot represents element-wise multiplication.

The GNN based object encoder is supervised by a sparse location loss, dense feature loss, and object matching loss [5]. The objective of the sparse location loss is to ensure that

¹Refer to [28] for more details on GNNs and GAT.

location feature x_i^L is a sparse vector. The sparse location loss L_s is defined as the l_1 -norm of x_i^L .

$$L_s = \sum_{i=1}^M \|\phi(x_i^L)\|_1, \quad (4)$$

where $\phi(x) = x/\|x\|_2$ is a l_2 -normalization to prevent the location features from being zero. Given that the sparse location loss ensures that keypoints are encoded into sparse locations on the object descriptor, the objective of the dense feature loss is to ensure that distinctive keypoints are encoded to unique sparse locations on the object descriptor. Hence, dense feature loss L_d is defined as the negative l_1 -norm of the location features.

$$L_d = \max \left(0, \delta - \phi \left(\left\| \sum_{i=1}^M (x_i^L) \right\|_1 \right) \right), \quad (5)$$

where $\delta > 0$ is a positive constant. Intuitively, the combined optimization of both sparse location loss and dense feature loss enables the object encoder to encode graph representations such that the similar keypoints are encoded to similar locations, while distinctive keypoints cover different locations retaining the density of the object descriptor.

Finally, to enable robust object matching, a triplet style object matching loss is used. The objective of the object matching loss L_m is to maximize the cosine similarity of positive object pairs and minimize the cosine similarity of negative object pairs.

$$\begin{aligned} L_m = & \sum_{\{p,q\} \in P^+} (1 - S(D_p, D_q)) \\ & + \sum_{\{p,q\} \in P^-} \max(0, S(D_p, D_q) - \zeta), \end{aligned} \quad (6)$$

where $\zeta = 0.2$, S is the cosine similarity, and P^+ and P^- are positive and negative object matching pairs, respectively.

C. Topological Graph Merging and 3D Object Descriptors

Given a sequence of temporal topological graph-based object representations, we use the sparse location features from the object encoder to build an evolving topological graph representation of the object. We believe that the properties of the sparse location features enable us to robustly match the keypoints of objects across various frames. Based on this intuition, we calculate the cosine similarity between location features of different frames and define a matching threshold of $\lambda = 0.9$ to select similar points. Then, based on this matching, new points in the current frame that didn't match any points will be appended to the original graph. To effectively merge the graph structure, we employ a union across the adjacency matrices of the merged frames based on the matching pairs to generate a graph structure (topology) for the incrementally evolving graph.

Since the graph structure remains constant within the object encoder, we take the temporally evolving topological graph representation based on the location features and build a similar graph with the same topological structure; however, the nodes are based on SuperPoint features. This SuperPoint

based evolving topological object graph representation is input to the object encoder to get the 3D object descriptor. We call this object descriptor a 3D object descriptor because it's an encoding based on a temporally evolving graph. So essentially, here, the third dimension is time as opposed to space. We believe that this 3D object descriptor reasons about varying object structure due to deformation or occlusion, enabling robust matching compared to a single-frame object descriptor.

IV. EXPERIMENTS

A. Implementation Details & Evaluation Criteria

The object encoder is trained on the COCO dataset [33], where the object-wise grouped features are obtained using ground-truth instance segmentations and pre-trained SuperPoint [7]. Furthermore, random homographies, including the translation, rotation, perspective, and scale transforms, are generated for data augmentation. For training, a batch size of 16, a learning rate of 10^{-5} with RMSprop [34] optimizer is employed. For the experiments, the particular configurations of the object encoder are set as $N_p = 256$, $N_m = 16$, and $N_o = 2048$.

To test the performance of our proposed framework, we use the YouTube Video Instance Segmentation (VIS) dataset [35] which is a large-scale video sequence dataset containing a large object vocabulary and various challenging scenarios, including perceptually aliased animals/objects and people riding on vehicles. Furthermore, the dataset provides ground-truth instance segmentations for video sequences, making the experimentation more robust than using a Mask R-CNN [8] to generate instance segmentations.

For testing the proposed framework, we build 3D object descriptors from the first half of the video sequence and match them with the 3D object descriptors of the second half. To determine a match between an object pair, we compute the cosine similarity between the descriptors and define a matching threshold ρ . We use a matching threshold of $\rho = 0.8$ to calculate the Precision, Recall, and F1-Score. Furthermore, by varying the threshold ρ values, we generate precision-recall curves and calculate the area under the curves.

We use two baselines: 2D Baseline and 3D Baseline, to extensively compare the performance of our proposed approach. For the 2D Baseline, similar to [5], we match the single-frame descriptors of the first half of the video sequence to the second half. Furthermore, for the 3D Baseline, we match the descriptors obtained by averaging the single-frame descriptors of the first half of the video sequence to the descriptors obtained by averaging the single-frame descriptors of the second half.

B. Results & Discussion

Table I and Fig 4 show the performance of the proposed approach on the YouTube VIS dataset. We also present the performance of the two baselines.

TABLE I
PERFORMANCE COMPARISON ON YOUTUBE VIS DATASET.

Methods	Precision	Recall	F1-Score
2D Baseline	80.31	52.05	63.16
3D Baseline	92.51	39.63	55.49
<i>Ours</i> : 3D Object Encoding	75.89	75.02	75.46

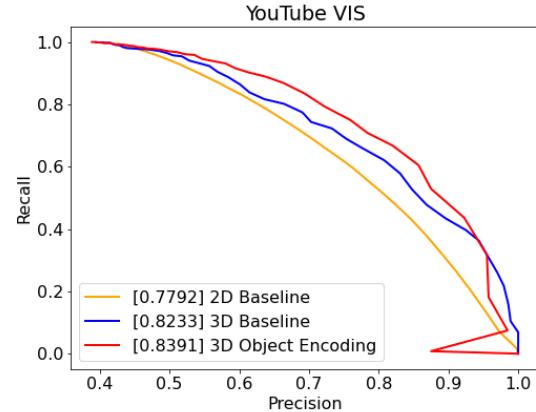


Fig. 4. Precision-Recall Curves for the proposed approach and baselines on YouTube VIS dataset. The area under the curve is shown in the [brackets].

From Table I, it can be observed that our method provides substantial performance improvements over the 2D and 3D Baselines. The proposed 3D Object Encoding based on evolving topological graphs shows an improvement in F1-Score of $\approx 12\%$ over the 2D Baseline and $\approx 20\%$ over the 3D Baseline. Furthermore, on observing the precision-recall curves shown in Fig 4, it can be seen that the proposed 3D Object Encoding has a higher area under the curve of $\approx 6\%$ compared to the 2D Baseline and $\approx 2\%$ compared to the 3D Baseline. Hence based on the performance trends, it can be observed that encoding an evolving graph representation of an object helps the 3D object descriptor to accumulate knowledge across multiple viewpoints and states, thereby providing more robust matching.

In Fig 5, we show qualitative visualizations of object matching for an aquarium video sequence in the YouTube VIS dataset, where 3D object descriptors based on evolving graph representations built till that current frame are matched with the single-frame object descriptors of the current frame. It can be seen that even though the video sequence poses challenging scenarios such as perceptual aliasing, deformation, and occlusion, the proposed method provides robust matching.

To further analyze the robustness of our proposed topological graph merging, in Fig 6, we provide visualizations of new distinctive features added to the evolving graph and also a plot showing the number of nodes in the evolving graph at different frames of a single-object boat video sequence in the YouTube VIS dataset. From Fig 6, it can be observed that new nodes are added to the evolving graph of the object only when new distinctive features of the object that

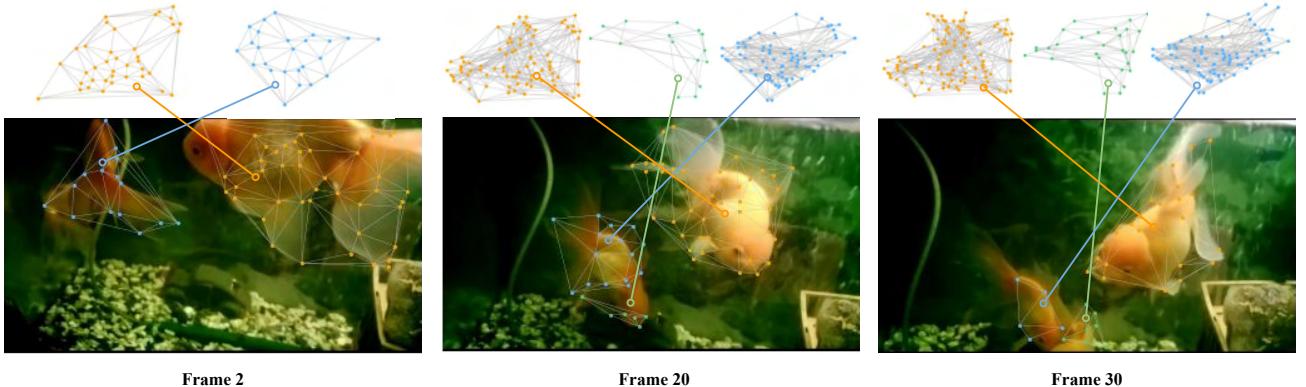


Fig. 5. Visualizations of object matching for a sequence in the YouTube VIS dataset. Here, we show matching between evolving graph representations of objects till the current frame and the single-frame topological graph representations of the current frame. Our method can robustly identify the fishes even though they look similar and showcase challenging properties such as deformation and occlusion.

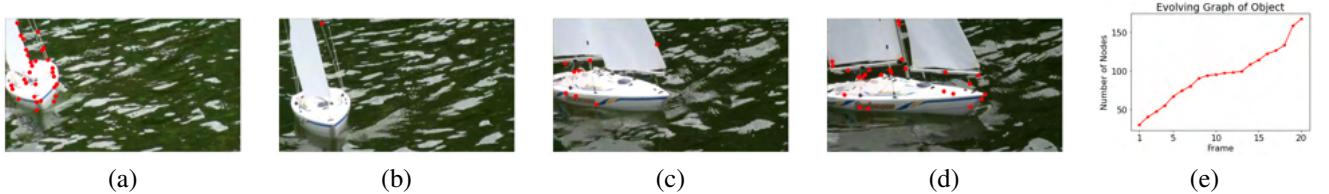


Fig. 6. (a) - (d) Visualizations of new distinctive features added to the evolving graph at the frames 1, 10, 16, & 19, respectively for a sequence in YouTube VIS, and (e) plot showing the number of nodes in the evolving graph at each frame of the same video sequence.

have not been seen a-prior are observed due to changes in viewing angles. This particular analysis shows our proposed topological graph merging method’s effectiveness in building incrementally evolving graphs where new nodes are added as distinctive features of the object are observed.

V. FUTURE WORK

In future work, we aim to explore training the object encoder for the task of generating 3D object descriptors. We currently use an object encoder trained for single-frame descriptors, and it would be interesting to explore an updated object encoder that is directly trained to generate 3D object descriptors. To facilitate this, we aim to formulate a differentiable topological graph merging method based on soft score assignment instead of the current hard threshold-based graph merging and supervise the graph merging process using SuperGlue with a formulated graph merging loss. Furthermore, it would be interesting to explore the concept of Cross-correlation matrices, which has been recently explored for Lifelong Graph Learning [36], in the context of our work.

We also aim to deploy the proposed framework in a simulated multi-agent distributed setup to test the effectiveness of 3D Object Encoding by building and sharing evolving graph-based representations in a distributed setup. Furthermore, it would be interesting to compare our proposed approach’s semantic loop closure performance in a distributed setup with recent state-of-the-art multi-agent loop closure methods based on image feature matching used in systems such as Kimera-Multi [6].

VI. CONCLUSION

Discriminatively identifying objects in a distributed setup is a critical and challenging problem for multi-agent robotics tasks involving autonomous exploration and semantic localization & mapping. It is crucial to build “evolving” object representations that accumulate knowledge across an object’s varying states in this context. In this paper, we present a novel evolving topological graph-based 3D object encoding method to generate global object descriptors. We propose a topological graph merging method that identifies new distinctive features robustly to build evolving topological graph representations. The experiments show that our proposed method leads to superior performance in object identification and semantic loop closure on a challenging dataset. We also show that our generated object descriptors are robust to drastic viewpoint shift, occlusion, deformation, and scale of the representation. We envision our proposed method to play an essential role in multi-agent frameworks used for robotic applications.

ACKNOWLEDGMENT

This work is supported by the Air Lab at the Robotics Institute, Carnegie Mellon University. The authors would like to thank Rachel Burcin, Dr. John Dolan, the CMU Robotics Institute Summer Scholars (RISS) program organizers, and sponsors for making this program possible.

REFERENCES

- [1] A. Sharma, W. Dong, and M. Kaess, “Compositional scalable object slam,” *arXiv preprint arXiv:2011.02658*, 2020.

- [2] C. Wang, W. Wang, Y. Qiu, Y. Hu, and S. Scherer, "Visual memorability for robotic interestingness via unsupervised online learning," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 52–68.
- [3] C.-C. Wang, C. Thorpe, S. Thrun, M. Hebert, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *The International Journal of Robotics Research*, vol. 26, no. 9, pp. 889–916, 2007.
- [4] M.-P. Dubuisson and A. K. Jain, "A modified hausdorff distance for object matching," in *Proceedings of 12th international conference on pattern recognition*, vol. 1. IEEE, 1994, pp. 566–568.
- [5] K. Xu, C. Wang, C. Chen, and W. Wu, "A robust object encoding method," *arXiv preprint arXiv:2105.00327*, 2021.
- [6] Y. Tian, Y. Chang, F. H. Arias, C. Nieto-Granda, J. P. How, and L. Carlone, "Kimeramulti: Robust, distributed, dense metric-semantic slam for multi-robot systems," *arXiv preprint arXiv:2106.14386*, 2021.
- [7] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superpoint: Self-supervised interest point detection and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 224–236.
- [8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.
- [9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [10] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.
- [11] A. Glover, W. Maddern, M. Warren, S. Reid, M. Milford, and G. Wyeth, "Openfabmap: An open source toolbox for appearance-based loop closure detection," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 4730–4735.
- [12] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2564–2571.
- [13] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.
- [14] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimeram: an open-source library for real-time metric-semantic localization and mapping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1689–1696.
- [15] E. Garcia-Fidalgo and A. Ortiz, "ibow-lcd: An appearance-based loop-closure detection approach using incremental bags of binary words," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3051–3057, 2018.
- [16] D. Schlegel and G. Grisetti, "Hbst: A hamming distance embedding binary search tree for feature-based visual place recognition," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3741–3748, 2018.
- [17] M. Gehrig, E. Stumm, T. Hinzmann, and R. Siegwart, "Visual place recognition with probabilistic voting," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3192–3199.
- [18] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International conference on computer vision*. Ieee, 2011, pp. 2548–2555.
- [19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.
- [20] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, "Deep learning features at scale for visual place recognition," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3223–3230.
- [21] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5297–5307.
- [22] S. Hausler, A. Jacobson, and M. Milford, "Multi-process fusion: Visual place recognition using multiple image processing methods," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1924–1931, 2019.
- [23] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1. Ieee, 2005, pp. 886–893.
- [24] M. Schwarz, H. Schulz, and S. Behnke, "Rbg-d object recognition and pose estimation based on pre-trained convolutional neural network features," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 1329–1335.
- [25] H. F. Zaki, F. Shafait, and A. Mian, "Convolutional hypercube pyramid for accurate rgb-d object category and instance recognition," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1685–1692.
- [26] ———, "Viewpoint invariant semantic object and scene categorization with rbg-d sensors," *Autonomous Robots*, vol. 43, no. 4, pp. 1005–1022, 2019.
- [27] P.-E. Sarlin, D. DeTone, T. Malisiewicz, and A. Rabinovich, "Superglue: Learning feature matching with graph neural networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 4938–4947.
- [28] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.
- [29] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 716–12 725.
- [30] N. Veetha, M. J. Milford, and S. Garg, "A hierarchical dual model of environment-and place-specific utility for visual place recognition," *IEEE Robotics and Automation Letters*, 2021.
- [31] M. J. Tarr and W. G. Hayward, "The concurrent encoding of viewpoint-invariant and viewpoint-dependent information in visual object recognition," *Visual Cognition*, vol. 25, no. 1-3, pp. 100–121, 2017.
- [32] D.-T. Lee and B. J. Schachter, "Two algorithms for constructing a delaunay triangulation," *International Journal of Computer & Information Sciences*, vol. 9, no. 3, pp. 219–242, 1980.
- [33] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [34] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent," *Cited on*, vol. 14, no. 8, p. 2, 2012.
- [35] L. Yang, Y. Fan, and N. Xu, "Video instance segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5188–5197.
- [36] C. Wang, Y. Qiu, and S. Scherer, "Lifelong graph learning," *arXiv preprint arXiv:2009.00647*, 2020.

Solving CommonRoad Motion Planning Benchmarks using CILQR

Shivesh Khaitan¹, Qin Lin² and John M. Dolan²

Abstract—Self-driving cars have seen great technological advancements in recent decades. Motion planning in dynamic environments is a critical component in the self-driving technology stack. However, unlike the perception and trajectory prediction tasks, the lack of real traffic benchmarks imposes a barrier for the research community to comprehensively validate and compare different motion planning algorithms. In this paper, we demonstrate how CILQR, which is an optimization-based algorithm, coupled with a high-level trajectory planner and a prediction framework, can be used to solve motion planning for self-driving in generic scenarios. To demonstrate the effectiveness of our framework in real-world scenarios, we test its performance using CommonRoad benchmark scenarios which have been created using real-world traffic datasets, e.g., the NGSIM dataset. Our framework is able to successfully solve a multitude of benchmark scenarios.

Index Terms—Autonomous Vehicle Navigation, Collision Avoidance, Motion and Path Planning, Optimization and Optimal Control

I. INTRODUCTION

Self-driving cars are at the heart of academia and industry due to the immense benefits which they can bring to society, ranging from reduced accidents to solving traffic jams and eliminating drivers' discomfort during long journeys. They have been around for a decade and technological advances are fueling further developments in the field. Research in the field has been promoted by several key industry leaders, e.g., Waymo and Tesla. Major events like the DARPA challenge, Hyundai Autonomous Challenge and the latest Indy Autonomous Challenge have also contributed to the progress. Most approaches to solving self-driving have divided the problem into subsystems, including mapping and localization, perception, and planning and controls.

Motion Planning is one of the most critical tasks in the development of driverless car technology. This is because motion planners need to deal with obstacles, road constraints, traffic rule constraints and other human drivers in the environment simultaneously while ensuring that the planned trajectory is dynamically feasible for the low-level controller to achieve. The highly dynamic environments pose a major challenge to decision making and planning in real-world driving scenarios. Though there have been several advancements in the field of motion planning over the years, solving motion planning for self-driving in complex traffic

scenarios is still a challenge which will require further work to be solved.

Previously, several motion planning frameworks have been proposed for self-driving. However, a majority of the existing solutions have been tested in limited, artificially designed scenarios. This is because motion planners alone are not able to deal with general self-driving scenarios. They need processed obstacle data for prediction and reference goals to determine the behavior to be executed. Thus, the research question raised in our work is how a motion planner can be coupled with a high-level route planner and a prediction framework to solve CommonRoad Motion Planning benchmarks [1] which have been created using real-world traffic data, e.g., from the NGSIM dataset.

CommonRoad Benchmarks are a collection of composable benchmarks for motion planning on roads. Reproducibility and comparability are important advantages of the CommonRoad tests that make them suitable for testing motion planning algorithms.

Figure 1 shows an overview of the solution architecture. For each planning problem, CommonRoad provides a traffic scenario, goal region and a lanelet [2] network describing the road network architecture. The CommonRoad interface also provides the updated state of the ego-vehicle and dynamic obstacles at every time-step. Time is discretized with a small time-step dt in each planning problem. The proposed solution works in a receding horizon framework. For each planning step, the high-level route planner generates a series of lanelets to traverse in order to reach the goal region specified in the planning problem. Concurrently, the prediction framework also generates future trajectory predictions of the dynamic obstacles. The motion planner then uses the reference lanelets, predicted dynamic obstacles and the road boundary as constraints to generate a kinematically feasible trajectory for the car's controller starting from the ego-vehicle's instantaneous state.

The rest of this paper is organized as follows. Section II provides a review of existing related work. Section III explains the CommonRoad Motion Planning problem in detail. Section IV contains necessary background on key methodologies. Section V is about the formulation of the vehicle dynamic model, cost function and constraints design. Section VI presents the experimental results. The conclusion is in section VII.

II. RELATED WORK

Several motion planning frameworks have been proposed for self-driving. The algorithms are divided into four major categories, including 1) search-based methods such as state

¹Shivesh Khaitan is with the Department of Computer Science & Engineering, Manipal Institute of Technology, Manipal, Karnataka, India shivesh.khaitan@learner.manipal.edu

²Qin Lin and John M. Dolan are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, the USA {qinlin, jdolan}@andrew.cmu.edu

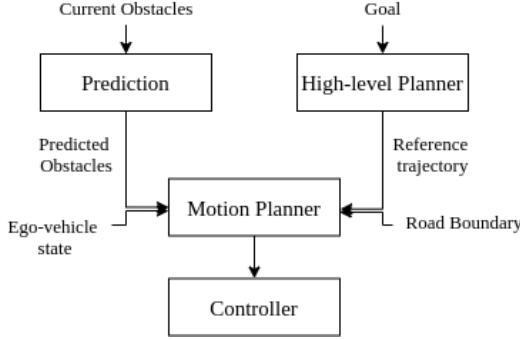


Fig. 1: Overview

lattice based planning [3] and Probabilistic Road Map (PRM) [4]; 2) sampling-based methods such as rapidly-exploring random trees (RRT) [5]; 3) various curve interpolation-based methods; 4) optimization-based approaches using sequential programming. End-to-end learning-based methods using Imitation Learning [6] and Reinforcement Learning [7] have also become popular for motion planning and decision-making in recent years.

Search- and sampling-based methods are often not suitable for highly cluttered urban scenarios due to their incomplete nature. Further, curve interpolation methods do not always guarantee kinematically feasible trajectories. Learning-based methods are still considered black-box solutions which require further study before they can be used in real-world scenarios.

Trajectory optimization-based approaches formulate the motion planning problem as a constrained optimization problem where real-world constraints like dynamics and collision avoidance are formulated as optimization constraints. Model Predictive Control (MPC) [8]–[10] is one of the widely used optimization-based approaches for planning and controls. Despite being highly popular, its use in real-world self-driving technology is limited due to high complexity of the self-driving car problem, leading to increased computation time. This complexity is mainly due to the highly nonlinear vehicle models and nonconvex free space in real-world driving scenarios, ultimately leading to nonlinear constraints in the optimization problem. Mixed-Integer Quadratic Programming (MIQP)-based solutions [11] have also been proposed, but again the nonlinear constraints pose a challenge due to high computation.

Linear Quadratic Regulator is a widely used method for optimal control of linear systems with quadratic cost function. Iterative linear quadratic regulator (ILQR) [12] is an extension of LQR for nonlinear systems which utilizes Dynamic Programming (DP), thus reducing computation compared to other nonlinear system solution methods. Constrained ILQR (CILQR) [13] is a further extension of ILQR which is able to deal with obstacle constraints by adding obstacles as quadratized cost in the objective function. However, most earlier approaches [13], [14] are limited to simple scenarios with static obstacles or vehicles moving with constant velocity.

In [15], the authors have further proposed a safe prediction framework for CILQR. However, the testing is still limited to a few specialized scenarios. In this work, we demonstrate a complete motion planning framework with route-planning and prediction for CILQR to solve real-world scenarios simulated using CommonRoad benchmarks.

III. COMMONROAD BENCHMARKS

Each CommonRoad benchmark consists of a scenario with a planning problem, a vehicle dynamics model, vehicle parameters, and a cost function composing a unique ID. The same scenario can be tested with different vehicle dynamics models, different vehicle parameters and different cost functions. The scenario selection tool available allows segregating specific kinds of scenarios, e.g. lane change, lane following, merging, traffic jam, turns, intersections, etc. A single planning problem can also have multiple scenario tests.

The ego-vehicle state space model is defined as:

$$\dot{x}(t) = f_M(x(t), u(t)) \quad (1)$$

where $x \in R^n$ is the state vector, $u \in R^m$ is the input vector and t represents time. For each planning problem, the following are defined:

- Initial time, t_0 : The solution trajectory must start from the initial time.
- Initial state, $x(t_0)$: The solution trajectory must start from the initial state.
- Lanelet network: The occupancy of the ego-vehicle for each state of the solution trajectory must lie within the lanelets.
- Dynamic obstacles: Occupancy of dynamic obstacles in the scenarios at each time-step should not have any overlap with the occupancy of the ego-vehicle at that time-step.
- Goal region, $\mathcal{G}_S \subseteq \mathbb{R}^n$: The solution's terminal state must be within the goal region.

The goal region can be a union of disjoint sets for different state constraints. Every goal region includes:

- Final time interval: $t_f \in [t_{f_{min}}, t_{f_{max}}]$. The solution trajectory must finish within the specified time interval.
- Final position: The final position can be specified in one of two ways: 1) Set of Lanelet IDs; 2) Convex polygon region. The center of the final position of the ego-vehicle must be within the specified lanelets / polygon.
- Final Orientation interval (*optional*): $\theta_f \in [\theta_{f_{min}}, \theta_{f_{max}}]$. The solution trajectory's final state's orientation must be within the specified orientation interval.
- Final Velocity interval (*optional*): $v_f \in [v_{f_{min}}, v_{f_{max}}]$. The solution trajectory's final state's velocity must be within the specified velocity interval.
- There can be additional constraints on yaw-rate, slip-angle, etc. depending on vehicle model. These are specified in a similar manner as intervals.

Apart from the vehicle dynamics (Eq. 1) and goal region constraints for the final state, there are additional constraints

which should not be violated for each of the solution trajectory states:

- The occupancy of the ego-vehicle should be within the drivable free-space (avoiding dynamic obstacles) for each time-step, i.e., $\forall t \in [t_0, t_f] : O(x(t)) \in \mathcal{W}_{free}(t)$ where $O(x(t))$ denotes the occupancy at time t .
- Control constraints, e.g., acceleration and steering limits.

IV. METHODOLOGY

In this section, we will go through key techniques used including route-planning, short and long-term prediction using reachability analysis and an adaptive filter and Constrained ILQR optimization.

A. High-level Route Planner

Route planning for generating a high-level plan to be followed by the motion planner is done using CommonRoad's inbuilt route planner. The route planner finds sequences of lanelets that lead from the initial lanelet to the goal lanelets of a given planning problem. The planner works by creating a directed graph of the lanelet network specified for the planning problem and running an A* [16] search in the graph. After generating the series of lanelets, it computes a reference path through the lanelets. However, the reference path might not be kinematically feasible and may have collisions with dynamic obstacles. These have to be dealt by the motion planner.

B. Prediction of dynamic obstacles

The state of the moving obstacles in the environment needs to be predicted in each planning loop to generate obstacle-free trajectories. This is achieved using the short-term and long-term prediction model from [15]. It proposes a combination of a safety-oriented short-term planner and an efficiency-oriented long-term planner.

The short-term prediction considers the uncertainty of a target vehicle's state (e.g., sensor disturbance or localization error) and the uncertainty of control actions over the short-term prediction horizon under a kinematically feasible but possibly non-deterministic assumption. The reachable state of the target vehicle is projected to the sub-space in the inertial frame for the min/max longitudinal and lateral positions. The long-term predictor only predicts the target vehicle's single position (i.e., particle) without considering uncertainty. For a detailed explanation of the short- and long-term predictions, readers are referred to [15].

C. Constrained Iterative Linear Quadratic Regulator (CILQR)

To compute the final trajectory, we make use of CILQR [13], [14]. An obstacle-free motion planning problem can be formulated as a standard ILQR problem with nonlinear system dynamics:

$$\min_{\mathbf{U}} \quad J = \sum_{k=0}^{N-1} l(\mathbf{x}_k, \mathbf{u}_k) + l_f(\mathbf{x}_N) \quad (2)$$

$$\text{s.t.} \quad \mathbf{x}_{k+1} = f(\mathbf{x}_k, \mathbf{u}_k) \quad (3)$$

where \mathbf{x}_k and \mathbf{u}_k are the state and the control input at time step k and \mathbf{x}_N is the final state. Thus, N is the planning horizon. Eq. 3 is the system dynamics constraint, which is a transition function mapping state and control at step k to state at step $k+1$. $\mathbf{U} := \{\mathbf{u}_0, \mathbf{u}_1, \dots, \mathbf{u}_{N-1}\}$ is the control sequence, and l and l_f are the cost functions.

Since the standard LQR only solves optimization problems with quadratic cost and linear systematic constraints, this problem can be reformulated. By linearizing the systematic constraint at multiple points, we can relax the nonlinearity of the ILQR problem into the linear problem required by LQR. The steps of ILQR are listed below.

- 1) Start with a feasible initial guess $\hat{\mathbf{u}}$ and obtain $\hat{\mathbf{x}}$ using the dynamic model. A common way is using a zero initialization. Note that feasibility of the initial guess is important in practice. The users can either do a sampling in the beginning or start the planning only when the zero initialization is feasible.
- 2) Calculate the derivatives of the dynamics and the cost function about the trajectory.
- 3) Run an LQR backward pass to obtain $\delta\mathbf{u}^*$. For an ill-conditioned matrix, we increase λ and restart the backward pass, otherwise we reduce λ . The details of designing appropriate factors of increasing and decreasing can be found in [17].
- 4) Run forward pass and initially set $\alpha = 1$ in $\delta\mathbf{u} = \alpha\mathbf{k} + K\delta\mathbf{x}$ to compute a new nominal sequence. If the cost does not converge, decrease α and restart the forward pass.

ILQR has the drawback of its constraint-free nature, which makes it unsuitable for collision avoidance problems. The CILQR algorithm offers the inclusion of different constraints into the objective function through barrier functions. Ideally, a barrier function serves as an indicator giving a huge penalty to constraint violation and low cost to satisfied constraints. Constraints can be generalized into two categories by linearity. First, any nonlinear constraints can be converted to linear constraints via a second-order Taylor Expansion. Then, a barrier function is applied and quadratized. Eq. 4 and Eq. 5 demonstrate this process. The quadratized linear barrier function can now be incorporated into the ILQR algorithm.

An exponential barrier function is defined as

$$b_k(g(\mathbf{x}_k)) = q_1 \exp(q_2 g(\mathbf{x}_k)) \quad (4)$$

Its Jacobian and Hessian are derived as

$$\begin{aligned} \nabla b &= q_1 q_2 \exp(q_2 g(\mathbf{x}_k)) \nabla g(\mathbf{x}_k) \\ \nabla^2 b &= q_1 q_2 \exp(q_2 g(\mathbf{x}_k)) (q_2 \nabla g(\mathbf{x}_k) \nabla g(\mathbf{x}_k)^T + \nabla^2 g(\mathbf{x}_k)) \end{aligned} \quad (5)$$

where $g(\mathbf{x})$ is the constraint function at time step k .

V. PROBLEM FORMULATION

In this section, we will go through the vehicle dynamic model, the cost function and constraints design.

A. System Dynamics

The vehicle model used is the kinematic single-track model (denoted as KS in CommonRoad) shown in Figure 2. It considers only two wheels, where the front and rear wheel pairs are each lumped into one wheel, because the roll dynamics are neglected. The tire slip is neglected as well. The differential equations of the vehicle model are defined as:

$$\begin{aligned}\dot{p}_x &= v \cos(\theta) \\ \dot{p}_y &= v \sin(\theta) \\ \dot{v} &= a_{long} \\ \dot{\theta} &= \frac{v}{L} \tan(\delta) \\ \dot{\delta} &= v_\delta\end{aligned}\quad (6)$$

where p_x and p_y are the position in the Cartesian plane

v is the velocity

θ is the orientation

a_{long} is the longitudinal acceleration

δ is the steering angle

v_δ is the steering velocity

L is the wheelbase.

Following are the constraints for the vehicle model:

$$\begin{aligned}v_\delta &\in [v_{\delta_{min}}, v_{\delta_{max}}] \\ v &\in [v_{min}, v_{max}] \\ \delta &\in [\delta_{min}, \delta_{max}] \\ a_{long} &\in [-a_{max}, a_{max}] \\ \sqrt{a_{long}^2 + (v\dot{\theta})^2} &\leq a_{max}\end{aligned}\quad (7)$$

The control input is $\mathbf{u} = [a_{long}, v_\delta]^T$.

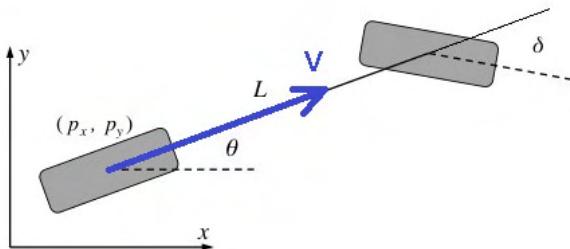


Fig. 2: Vehicle Kinematic Bicycle Model

B. Objective Function for CILQR

The general definition of the objective function in Eq. 2 can be made specific in Eq. 8. The terms in the summation represent the control effort cost, the reference position tracking, the reference velocity tracking, the reference orientation tracking and the constraints cost (i.e., obstacle avoidance). c_N is the end state cost.

$$J = \sum_{k=0}^{N-1} \left(c_k^u + c_k^{ref} + c_k^{vel} + c_k^\theta + c_k^{con} \right) + c_N \quad (8)$$

1) *control effort cost*: The penalty for large acceleration and the steering speed with corresponding weights are shown in Eq. 9.

$$c_k^u = \mathbf{u}_k^T \begin{bmatrix} w_a & \\ & w_{v_\delta} \end{bmatrix} \mathbf{u}_k \quad (9)$$

2) *position, velocity and orientation tracking cost*: The reference tracking term assigns a cost based on the distance to the closest point of the reference trajectory. The velocity cost penalizes the ego-vehicle for the difference between its velocity and the reference velocity. Similarly orientation cost penalizes for deviation from the reference orientation. The combined cost is written in a matrix form in Eq. 10, where $\Delta \mathbf{x}_k$ is the difference between the ego-vehicle state and the reference state.

$$\begin{aligned}\Delta \mathbf{x}_k &= \mathbf{x}_k - \begin{bmatrix} p_{x,k}^{ref} & p_{y,k}^{ref} & v_k^{ref} & \theta_k^{ref} \end{bmatrix}^T \\ c_t^{ref} + c_t^{vel} + c_t^\theta &= \Delta \mathbf{x}^T \begin{bmatrix} w_{ref} & & & \\ & w_{ref} & & \\ & & w_{vel} & \\ & & & w_\theta \end{bmatrix} \Delta \mathbf{x}\end{aligned}\quad (10)$$

The reference path from the route planner is a series of points in the Cartesian space which does not include any time or velocity reference. Since the environments are dynamic and the solution trajectory is required to be within the goal region in the specified time-interval, planning needs to consider both space and time dimensions. Further, since motion planning can only be done for a short horizon, intermediate goals need to be generated.

Thus, for each planning iteration, a local reference is computed. The reference path from the route planner is resampled at turnings to decrease the distance between sampled points. This is because at turnings, vehicle velocity needs to be less than the usual velocity. Thus, instead of setting v_k^{ref} explicitly, we implicitly add it using the distance between reference points. Hence, w_{vel} is set zero throughout except when the goal region is within the planning horizon. In this case, the v_k^{ref} is set to the average of the required velocity interval as defined in the planning problem and w_{vel} is set to be nonzero. For orientation as well, w_θ is set to zero unless the goal region is within the planning horizon.

The time dimension is added to ensure that the ego-vehicle reaches the goal region in the specified time interval. It is done as follows:

1) The time remaining is calculated as

$$t_r = t_{f_{avg}} - t_k$$

where $t_{f_{avg}}$ is calculated as

$$\begin{aligned} t_i &= \max\left(\frac{t_{f_{min}} + t_{f_{max}}}{2}, t_{f_{max}} - 5\right) \\ t_{f_{avg}} &= \begin{cases} t_i, & \text{if } t_i > t_k \\ t_f, & \text{otherwise} \end{cases} \end{aligned} \quad (11)$$

2) The reference path, after resampling for turnings, is divided into $\lceil \frac{t_r}{dt} \rceil$ segments such that each of the segments has an equal number of reference path points (the number of points in the last segment can be adjusted if t_r is not exactly divisible by dt). The centers of the first N segments are set as the position reference for CILQR.

The end state cost is set in a similar manner. The penalty cost for the end state is set much higher than for the intermediate states. This is to ensure that the ego-vehicle always prioritizes obstacle avoidance over reference tracking and at the same time moves in the general direction of the goal instead of strictly following the reference path, which might lead to collisions.

3) *constraint cost*: All inequality constraints can be expressed in a negative null form shown in Eq. 12, in which x_{lim} is the maximum or minimum boundary value and $f(x)$ is some sort of function on the decision variable.

$$g(x) = x_{lim} - f(x) \leq 0 \quad (12)$$

For linear constraints like acceleration and steering velocity limits, we can write them as, for instance:

$$g(u) = u - u_{max} \leq 0 \quad (13)$$

Then, a barrier function can be used as stated in Eq. 4.

For the obstacle avoidance term, we use a geometric collision check as the inequality constraint and the problem is formulated as a geometry-based cost function. Obstacles are formulated as ellipses with major and minor axes adjusted for the ego-vehicle's shape. The inequality constraint is shown in Eq. 14. θ_k^{obs} is the predicted heading angle of the obstacle at time k , and b are the semi-major and -minor axes' lengths. Δx_k and Δy_k are the relative longitudinal and lateral distance between the ego-vehicle and the predicted obstacle, respectively. The corresponding Jacobian and the Hessian of this term for the LQR backward pass can be found in Eq. 5.

$$\begin{aligned} R_k &= \begin{bmatrix} \cos(\theta_k^{obs}) & -\sin(\theta_k^{obs}) \\ \sin(\theta_k^{obs}) & \cos(\theta_k^{obs}) \end{bmatrix} \\ T_k &= R_k \begin{bmatrix} \frac{1}{a^2} & 0 \\ 0 & \frac{1}{b^2} \end{bmatrix} R_k^T \\ g([\Delta x_k, \Delta y_k]) &= 1 - [\Delta x_k, \Delta y_k] T_k [\Delta x_k, \Delta y_k]^T \leq 0 \end{aligned} \quad (14)$$

4) *road boundary cost*: To ensure that the CILQR does not violate the road boundary constraints, the road boundary is included in a manner similar to the obstacle cost for dynamic obstacles. However, the road boundaries need to

be incorporated differently, as they are not point obstacles. Algorithm 1 describes the obstacle generation.

Algorithm 1 Road Boundary Constraints

Compute (L)

inputs : L , list of lanelets in the network

output : O , computed list of obstacles

$O \leftarrow \emptyset$

foreach lanelet $l \in L$ **do**

if $l.left \subset \emptyset$ **then**

$l.left \leftarrow l.left \cup l.left_boundary$

if $l.right \subset \emptyset$ **then**

$l.right \leftarrow l.right \cup l.right_boundary$

return O

As described in algorithm 1, for each of the lanelets, it is checked whether there exists an adjacent lanelet to its left (or right) or not. If there is no lanelet to its left (or right), this means that the area on the left (or right) is not drivable free-space. Then, the obstacles list is appended with the sampled boundary points on the left (or right). Otherwise there exists an adjacent lane to the left (or right) and there is no need for any boundary constraint. The boundary points added to the obstacles lists are then added as costs in a manner similar to the dynamic obstacles. The radius of the obstacles is set as the distance between the sampled boundary points to form a circle. Since the obstacle is a circle, orientation does not have any effect and θ_k^{obs} is set to zero.

VI. EXPERIMENTAL RESULTS

In this section, we will present the experimental results we have for CommonRoad benchmarks. The ego-vehicle model used is kinematic single-track with vehicle parameters of a Ford Escort. We have tested using the TR1 cost function. The cost function penalizes the solution for longitudinal jerk (J_{long}), steering rate (SR), distance to obstacles (D) and deviation from lane center (LC). Thus we solve the scenarios in the setting KS1:TR1:. The scenarios for testing are taken from the CommonRoad's Competition for Motion Planning of Autonomous Vehicles 2021 - Phase 1¹.

Fig. 3 shows five of the scenarios which were used for testing. Due to limited space, we have shown four snapshots for each scenario beginning with the initial state and ending at the final goal state. The ego-vehicle is pictured in red. The actual trajectory of the ego-vehicle is pictured in dark blue. Dynamic obstacles (including vehicles, buses, pedestrians etc.) are depicted in light blue. The yellow-regions are the goal lanelets. The grey areas are the road network with lane markings in black. All the scenarios are in the noninteractive mode of CommonRoad. Thus, the surrounding vehicles do not cooperate with the ego-vehicle.

The multitude of scenarios types for which the testing was conducted is evident in the figures.

- (a) DEU_Flensburg-1_1_T-1: Sharp right turn with vehicle behind.

¹<https://commonroad.in.tum.de/competition/announcement>

- (b) ZAM_Tjunction-1_48_T-1: Busy T-junction.
- (c) DEU_Flensburg-60_1_T-1: Different vehicles including bus, bicycle and pedestrian.
- (d) DEU_Lohmar-39_1_T-1: Urban traffic
- (e) DEU_Flensburg-71_1_T-1: Narrow road with single lane

We have used a planning horizon of 4 seconds with a discretization (dt) of 0.1 seconds. The average loop time for the framework is less than 300ms in the Python implementation. Thus, we expect a real-time performance in C++. The framework runs on a laptop with a 2.50GHz Intel Core i5-7200U CPU.

VII. CONCLUSION

In this paper, a framework for solving the CommonRoad's Motion Planning Benchmarks has been proposed. The framework is able to successfully generate solutions in different kinds of scenarios including urban, junctions, turns etc. We demonstrate how CILQR, coupled with a route planner and prediction framework can be used to generate kinematically feasible obstacle free trajectories for self-driving in various situations.

Further work in this approach includes extending the approach to test with the highly complex vehicle models and improving the implementation for real-time performance. Another area of research would be to improve the prediction framework using the road network constraints available. Integrating a robust behaviour planner to decide intermediate goals would also improve the performance.

VIII. ACKNOWLEDGMENT

This work was supported by the CMU Robotics Institute Summer Scholars (RISS) Program. Shivesh Khaitan would also like to thank Dr. John M. Dolan and Ms. Rachel Burcin for their constant support.

REFERENCES

- [1] M. Althoff, M. Koschi, and S. Manzinger, "Commonroad: Composable benchmarks for motion planning on roads," in *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 719–726.
- [2] P. Bender, J. Ziegler, and C. Stiller, "Lanelets: Efficient map representation for autonomous driving," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, 2014, pp. 420–425.
- [3] J. Ziegler and C. Stiller, "Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 1879–1884.
- [4] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.
- [5] S. M. LaValle *et al.*, "Rapidly-exploring random trees: A new tool for path planning," 1998.
- [6] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4693–4700.
- [7] C. Paxton, V. Raman, G. D. Hager, and M. Kobilarov, "Combining neural networks and tree search for task and motion planning in challenging environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 6059–6066.
- [8] Y. Gao, T. Lin, F. Borrelli, E. Tseng, and D. Hrovat, "Predictive control of autonomous ground vehicles with obstacle avoidance on slippery roads," in *Dynamic systems and control conference*, vol. 44175, 2010, pp. 265–272.
- [9] A. Arab, K. Yu, J. Yi, and D. Song, "Motion planning for aggressive autonomous vehicle maneuvers," in *2016 IEEE International Conference on Automation Science and Engineering (CASE)*. IEEE, 2016, pp. 221–226.
- [10] F. Borrelli, P. Falcone, T. Keviczky, J. Asgari, and D. Hrovat, "Mpc-based approach to active steering for autonomous vehicle systems," *International journal of vehicle autonomous systems*, vol. 3, no. 2-4, pp. 265–291, 2005.
- [11] X. Qian, F. Altché, P. Bender, C. Stiller, and A. de La Fortelle, "Optimal trajectory planning for autonomous driving integrating logical constraints: An miqp perspective," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016, pp. 205–210.
- [12] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems," in *ICINCO (1)*. Citeseer, 2004, pp. 222–229.
- [13] J. Chen, W. Zhan, and M. Tomizuka, "Constrained iterative lqr for on-road autonomous driving motion planning," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–7.
- [14] J. Chen, W. Zhan, and M. Tomizuka, "Autonomous driving motion planning with constrained iterative lqr," *IEEE Transactions on Intelligent Vehicles*, vol. 4, no. 2, pp. 244–254, 2019.
- [15] Y. Pan, Q. Lin, H. Shah, and J. M. Dolan, "Safe planning for self-driving via adaptive constrained ilqr," in *2020 International Conference on Intelligent RObots and and Systems (IROS)*. IEEE, 2020.
- [16] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.
- [17] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 4906–4913.

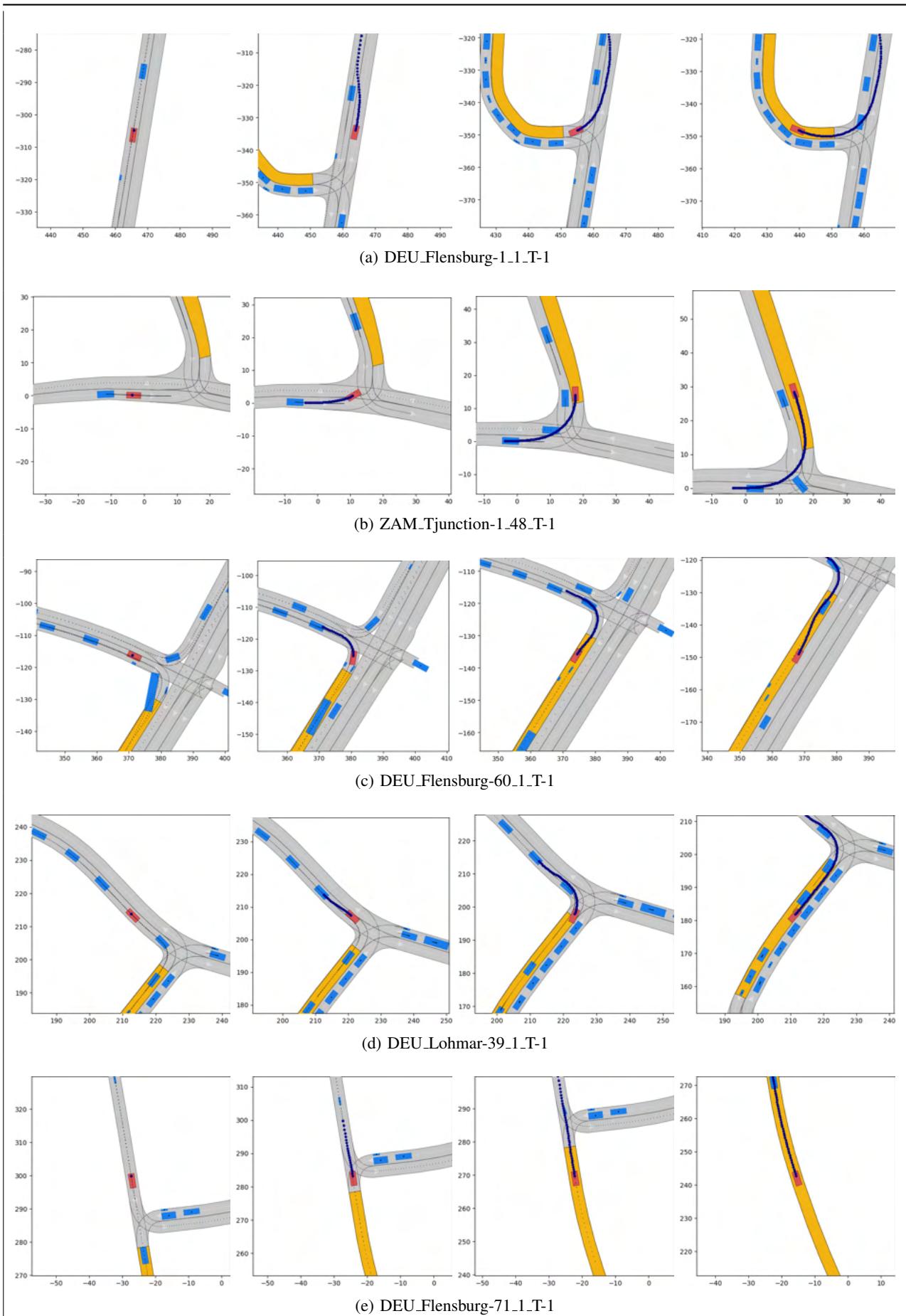


Fig. 3: Scenarios

Modelling Human Trust in Commanding of Robot Wingmen

Manav Kulshrestha¹, Huao Li², Konstantinos Mitsopoulos³, Dana Hughes⁴, Katia Sycara⁵

Abstract—In the development of AI systems that interact with humans and help them achieve a common goal, trust plays a key role in the efficient utilization of the communicated information. Simply put, for peak effectiveness, a human must be able to reasonably trust the information it gets from their AI counterpart. In working towards the same goal, the AI counterpart is similarly incentivised to maintain a good reputation with regards to its reliability as a teammate. We explore these ideas by allowing the AI to model its understanding of this trust and adapt accordingly as it performs in a 2-dimensional game environment with multi-faceted goals. This model aims to supplement the AI’s performance by allowing it to better predict human intentions and capabilities throughout the simulation in order to be a better wingman to the human pilot. The scope of the project will also extend to how the dynamics of the situation evolve with multiple such AI wingmen.

I. INTRODUCTION

The growing ubiquity of automation has seen advanced robots appear in a variety of work environments, many of which see them working alongside humans. Human presence is quite important in applications where there is a common task since humans can better adapt to changing mission guidelines and tasks, especially when the robots are built to be less specialized in their implementation. Our scenario deals with an aerial simulation environment. For tasks in this vein, humans are present in the form of a commanding vessel which is aided by some amount of UAVs in a supplementary capacity which may be responsible for environmental monitoring, exploration, or aiding in defensive formations. In an abundance of cases, humans may find themselves depending on the robots in very difficult situations with incredibly high stakes. Unfortunately, in the field of machine learning, the progressing sophistication of the automata’s decision processes often sees an incomprehensible obfuscation as the model for their mind learns and adapts to more complicated situations and tasks. This distancing of the human’s understanding of the inner workings of the automaton can have a negative impact on their ability to trust the information being provided by the automaton. In addition to this, the limited understanding also typically extends to accurately understanding the results from the UAVs as well as knowing the inputs that lead to the desired behaviors

¹Manav Kulshrestha is an undergraduate student with the College of Information and Computer Sciences, University of Massachusetts at Amherst

²Huao Li is a Ph.D. student with the Intelligent Systems Program, University of Pittsburgh

³Konstantinos Mitsopoulos is an Associate Project Scientist with the Robotics Institute, Carnegie Mellon University

⁴Dana Hughes is a Postdoctoral Fellow with the Robotics Institute, Carnegie Mellon University

⁵Katia Sycara is a Research Professor with the Robotics Institute, Carnegie Mellon University

which directly affects effective command the UAV fleet. All of these factors foster varying levels of trust depending on the human operator’s interpretation, causing under-trusting of information which may be vital to the success of an operation or the over-trusting of information with questionable veracity. Another aspect of this problem is that providing an abundance of information can have a detrimental impact on the effectiveness of humans, especially in situations where time is of the essence. In the case that the human under-trusts the UAVs, they may micromanage by being unnecessarily precise in their instruction or by taking over tasks that could have been delegated. In the case that the human over-trusts the UAVs, they may blindly accept the recommendations from the UAVs and fail to accurately monitor the situation. To summarize, trust plays a very important role in operations involving human-automaton cooperation and requires the automata to have a good sense of the human’s trust in order to adapt its actions with the goal of improving overall performance of the group.



Fig. 1. Environment concept for loyal wingman scenario

A. Markov Decision Processes

A Markov Decision Process (MDP) is represented as a tuple $\langle S, A, R, T, \gamma \rangle$. Here, S denotes the set of possible states the agent can be in. The state of the environment at time t is given by the random variable S_t . A denotes the set of actions that the agent can select between. The action chosen by the agent at time t is denoted by the random variable

A_t . $R : S \times A \times S \rightarrow \mathbb{R}$ is a reward function and characterizes the distribution over rewards at time t given by S_t, A_t , and S_{t+1} . $T : S \times A \times S \rightarrow [0, 1]$ is a state transition function and characterizes the distribution over states at time $t+1$ given the state and action at time t . An important factor is the Markov property which states that the distribution over S_{t+1} is independent of all prior events given S_t and A_t . That is to say that the distribution over states at time $t+1$ is fully determined by the state and action at time t . Finally, $\gamma \in [0, 1]$ is the reward discount parameter which is used to discount rewards based on how far in the future they occur. $\pi : S \rightarrow A$ is a policy which essentially specifies the distribution over A_t given S_t . The main problem concerning MDPs is to find the optimal which maximizes the cumulative reward function given by

$$\sum_{k=1}^{\infty} \gamma^k R(s_k, a_k, s_{k+1})$$

B. Inverse Reinforcement Learning

The Inverse Reinforcement Learning (IRL) problem is to find a reward function that best explains the observed behavior. In the simplest of cases, the state space would be finite, the model would be known, and the complete policy would be observed. In that, we would be given a finite state space S , the set of all actions A , the transition function T , and the reward discount parameter, and a policy π . From here, we are to then find the set of all possible rewards such that π is the optimal policy for the MDP given by $\langle S, A, R, T, \gamma \rangle$. Next, when considering the case of infinite states, the MDPs can be defined in the same way as above. However, we allow an isomorphism $S = \mathbb{R}^n$ for the sake of concreteness. Here, the reward function is a function from \mathbb{R}^n to the reals (which trivially extends to include actions). From here, we can simply use a linear approximation for the reward function, expressing it as

$$R(s_k, a_k, s_{k+1}) = \sum_{j=1}^d \alpha_j \phi_j(s_k)$$

where all ϕ_j are fixed, known, and bounded basis mapping functions from $S = \mathbb{R}^n$ to \mathbb{R} and the α_j are unknown parameters we want to fit.

II. RELATED WORKS

J. D. Lee and See's (2004) closed loop model, the human operator would receive the information on the state of the automata's decision processes from a display [1]. The trust was then determined as a function of the automata's capability as well as the current state. To elaborate, their closed model of the human operator's trust had 6 stages. As per the automata's understanding, after receiving information from the display, the human operator would analyze said information and form their belief regarding the situation. This allows the trust to evolve which can either cause more analysis and belief formation or go on to formulate an intention. From here, there is the simple matter of action reliance. Sheridan (2019) expands on this by proposing a Kalman estimation model by developing a feed-forward influence of

trust anticipation which evolves trust after intentions have been formulated [2]. This essentially allowed the trust model to anticipate the changes in the system state as intervening decisions were made by the human operator. The continuous updating of the trust was achieved by using the discrepancy between the state of the model and what was displayed to the human operator as well as what was anticipated to be the effects of the intervention. On another note, Nam, Walker, Lewis, and Sycara (2017) proposed a computational trust model for a foraging task where a swarm of robots was controlled by a human operator and the goal was to search for hidden targets in an unknown environment with dynamic goals [3]. An interesting problem faced was that since the goals were dynamic, interventions by the human operator was not necessarily indicative of lowering trust. In order to resolve this, they developed a classifier to distinguish between intentional shift in priorities and loss in trust.

A. Online Probabilistic Trust Inference Model

Xu and Dudek (2015) proposed a dynamic Bayesian inference model which used the performance of the automata in order to predict the operator's trust and supplemented it by taking periodic trust reports from the human operator in the middle of the task as well as interpreting the human operator's intervening action or returning control to the automata [4]. OPTIMo treats the degree of human-robot trust t_k at each time step k as a random variable, and maintains belief distributions for these performance-centric trust measures, based on various factors of the interaction experience. This was achieved by creating a Bayesian network which also incorporated past time steps in order to influence the current trust level. This graph structure efficiently encodes both causal and evidential relationships between trust and other factors, as well as the evolution of trust states over time.

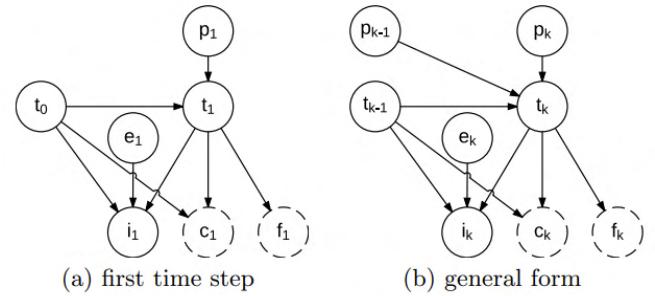


Fig. 2. Dynamic Bayesian structure of OPTIMo from Xu and Dudek (2015). Dashed factors are not observed on all time steps k , and are not mandatory for inferring trust states t_k

III. METHODOLOGY

Our experiment will utilize two different representations to model human trust with the intention of comparing and analyzing their accuracy, namely: inverse reinforcement learning and an online probabilistic inference model. The former sees human trust formalized as an MDP and will use IRL to learn the underlying reward function for the human

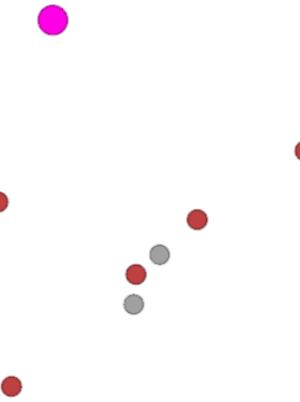


Fig. 3. Wingman scenario simulated in MAPE. 5 Targets (red), 2 UAVs (grey), and 1 Human operator (pink)

in an effort to learn the trust behavior. The latter is to use the Bayesian inference model which was detailed in Xu and Dudek (2015) [4]. We designed our experiment to simulate a two dimensional representation of an aerial environment. The environment will be populated by 1 human pilot, 2 unmanned aerial vehicles (UAVs) and 5 opposing targets. All targets can have a capability level of 0,1, or 2 whereas each of the UAVs as well as the human operators will have a capability level of 1. The basic idea behind defeating a target will be to engage it in either a group or alone so that the cumulative capability level is at least equal to that of the target. The overarching goal of the experiment, as a human participant, is to defeat targets and maximize the score while making effective use of the UAVs. The main challenge, from the human participant's perspective, is to try and gauge the reliability of each of the UAVs.

The format of the experiment is designed to have 20 trials where the properties of the UAVs remain persistent through the trials. That is, observed performance of each of the specific UAVs is expected to remain consistent. Each trial consists of two phases. Phase one will begin with each of the UAVs going out to survey a different target. The UAVs will then convey the information regarding the perceived capability level of the target with a confidence score. This confidence score will be obscured with some gaussian noise. Then, in the second phase, the human operator simply chooses to engage one of the 5 targets. Any successfully defeated targets will be counted towards the total score, weighted by the capability level of the defeated target. Finally, we ask the human operator for a subjective rating regarding their trust towards each of the two UAVs. From here, we simply reset the environment and repeat until the

participant has completed 20 trials.

For our experiments, we decided on different levels of reliability for UAVs and performed trials for each permutation of UAV configurations. The following details the multinomial distribution used for calculating the chance of the UAVs detected level matching the true level of the target.

True level	Detected level		
	Level 0	Level 1	Level 2
Level 0	0.8	0.1	0.1
Level 1	0.1	0.8	0.1
Level 2	0.1	0.1	0.8

TABLE I
MULTINOMIAL DISTRIBUTION FOR UAV TYPE 1

True level	Detected level		
	Level 0	Level 1	Level 2
Level 0	0.6	0.2	0.2
Level 1	0.2	0.6	0.2
Level 2	0.2	0.2	0.6

TABLE II
MULTINOMIAL DISTRIBUTION FOR UA TYPE 2

From the human perspective, there are several different considerations to be made with regards to the experiment. The operator has to build a continuous understanding of which UAVs it feels is more reliable and establish trust for. In addition to this, the addition of gaussian noise to the confidence levels of the readouts from the UAVs make it so the trust will not simply be a function of the confidence level.

IV. RESULTS AND CONCLUSIONS

Our research goal is to rate the accuracy of the trust rating modelled by the data from the UAVs with the ground truth being the one provided by the human participants during the trials. Another goal is to compare the performance of the two models, with respect to accuracy. The following is some data from Li et al. (2021) where a comparable experiment modelled trust using Kalman filters [5]. A notable exception was that the experiment involved supervisory control of swarms as opposed to having the human operator be participating in the field.

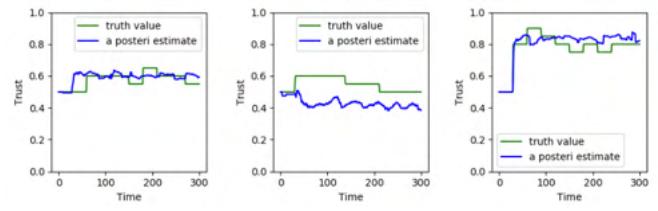


Fig. 4. Kalman estimation to model trust in Li et al (2021) [5]. The blue line represent model prediction while green is user trust feedback values

V. FUTURE WORK

Some next steps would include a second phase to the trials where the UAVs would coordinate with the human pilot in order to formulate the next steps which could take the form of the UAVs recommending possible plans for the human to either accept or reject. Naturally, an accept would indicate the building of trust whereas a rejection would indicate trust loss. Another idea would be to compare the performance of alternate models such as Kalman filters. A more ambitious extension of this project would be to use the data obtained from these experiments and embed them within the UAVs for a real time test along side human participants as opposed to discrete trials.

ACKNOWLEDGMENT

This work was funded by the National Science Foundation (NSF) under Grant No. 1659774. A special thanks to all the RISS mentors who assisted us along the way and helped make this possible including, but not limited to: Rachel Burcin, John Dolan, and Jennie Piotrzkowski.

REFERENCES

- [1] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," *Human factors*, pp. 50—80, 2004.
- [2] T. B. Sheridan, "Extending three existing models to analysis of trust in automation: Signal detection, statistical parameter estimation, and model-based control," *Human factors*, vol. 61, pp. 1162–1170, Feb. 2019.
- [3] C. Nam, P. Walker, M. Lewis, and K. Sycara, "Predicting trust in human control of swarms via inverse reinforcement learning," *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (ro-man)*, pp. 528–533, Aug. 2017.
- [4] A. Xu and G. Dudek, "Optimo: Online probabilistic trust inference model for asymmetric human-robot collaborations," *10th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, March. 2015.
- [5] H. Li, M. Lewis, and K. Sycara, "A kalman estimation model of human trust in supervisory control of robotic swarms," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 64, pp. 329–333, Feb. 2021.

Tuning Mechanical and Aerodynamic Properties of Micro Whisker Sensors for Improved Airflow Detection and Stimuli Sensing

Courage Lahban¹ Teresa Kent² Dr.Sarah Bergbreiter³

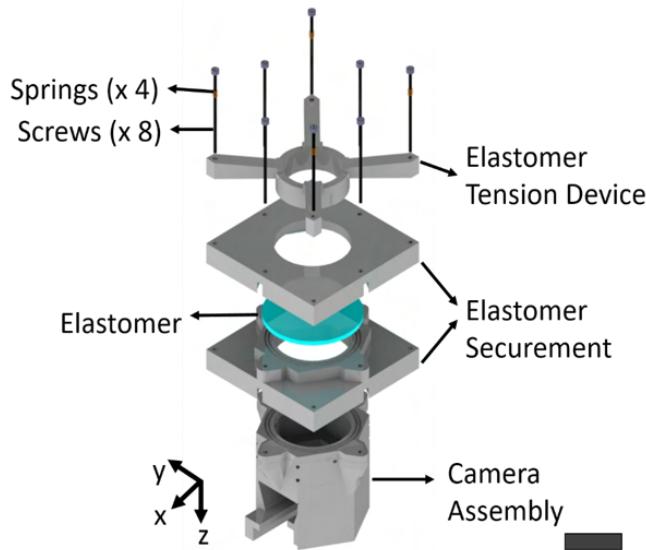


Fig. 1. Design of the WhiskSight sensor and primary components. This figure was first published in [6]

Abstract—Modifications to the WhiskSight sensor design and attached whiskers can amplify or mechanically filter properties of airflow applied to the WhiskSight sensor. Specifically, two criteria were varied: the layout of the whisker array, and mechanical properties of the elastomer substrate. Different array configurations will change the response to airflow. We hypothesize that the arrangement of the whisker array will affect the ability for all whiskers to detect airflow depending on the relative angle between the airflow and the array. Finally, the WhiskSight sensor uses an elastomer membrane to suspend the whiskers over a camera; changing the elastomer membrane's modulus of elasticity will affect both sensitivity and may even provide a mechanical filter for airflow. For example, we hypothesize that a low modulus elastomer will increase the angle of deflection and oscillations. The sensor will be characterized using a commercial fan and previously developed tracking algorithms. The methods used in the development and calibration of micro whisker sensors is discussed along with the experimental results

I. INTRODUCTION

Mammals can use their whiskers to trace their path by detecting flow features from their surroundings without necessarily using their visual senses. With a bilateral array arrangement of more than 25 whiskers located per side of

¹ Courage Lahban is with Department Mechanical and Industrial Engineering, New Jersey Institute of Technology Newark, NJ 07102 USA

² Teresa Kent and ³ Dr.Sarah Bergbreiter are with the Department of Mechanical Engineering, Micro Robotics Lab, Carnegie Mellon University, Pittsburgh, PA 15213 USA

the nose [1], mammals can use the whiskers to plan rapid motion ahead of their perception. Scientific studies reveal that seals can use their whiskers to locate their prey even when prevented from eye vision or sound sensing [9]. Engineered vibrissae (whiskers) can replicate the properties demonstrated by mammals to obtain multi-dimensional tactile information when applied in the surrounding environment to distinguish from contact, drag and inertia forces generated by the movements of the whisker motion [6].

Biologically inspired sensing for stimuli like those exhibited by mammals have been hypnotizing for implementation over a wide spectrum in robotics applications. Present in most mammals [2], whiskers augment vision by providing additional sensory information especially during full or partial occlusion of light. Active sensing in robotics can be used to enhance sensory systems like optical sensors and computer vision systems, which will improve the sensory capacity of mechanical systems such as mobile robots [3]. Direction sensing robots and autonomous systems can use this active sensing technique to read locations, detect speeds and control movements from lower pressure to higher pressure surroundings [4].

Different designs and approaches have been used for the development and optimization of whisker-based sensing systems. In a new design called “Touch the Wind” [4], a group of scientists use a whisker inspired high drag sensor to measure flow interaction and sensing dynamics for a Micro Aerial Vehicle (MAV) using whiskers. They adopt a deep learning and model-based approach to estimate UAV velocity interaction with the wind, measuring airflow at values up to 4 m/s [4].

Since airflow sensitivity is important for whisker performance, whisker sensing can also be used to detect and differentiate flow at more closer ranges when applied towards surface ranges. Another whisker sensor called the TacWhisker, uses a camera to track engineered whiskers during whisking to localize an object [5]. A non-similar approach incorporates magnetic hall effects sensors on rigid whiskers as trackers for tactile sensing [3, 4]. The highlighted approaches above show that current work on whiskers is largely inspired by the application and direct implementation, but whisker performance may vary based on the techniques and methods applied during development, and the purpose the whisker is been developed.

When it comes to whisker optimization, we are specifically interested in the general properties such as whisker oscillation and deflection which amount for the behavior and information obtained from the whisker sensors when applied under different flow conditions. In this paper, we explore different tuning techniques to optimize the performance of the WhiskSight sensor [6] for detecting specific variations in wind speed in an airflow range. We look at the mechanical properties of the

elastomer substrate and study how the distinction in information produced by different materials under airflow conditions may highlight new properties in the behavior of whiskers. Two variations were made: comparing material thickness and observing the whisker responses.

In addition, we seek to understand how whisker placement and orientation relative to airflow origination impacted average deflections and oscillations produced when flow is applied. Whisker based sensing systems face current limitations, first being the control of whiskers during active sensing which is important to amplify the signal in quality and quantity of sensing information obtained during sensing [7] and secondly, the improved multi-dimensional flow sensing and optimal performance when introduced to different material and array configurations [6]. Understanding the tuning properties of whiskers is important for future optimization in multi-dimensional stimuli detection using different materials and designs.

II. METHODS

Two different tuning experiments were performed. We investigate on the first test how different materials (elastomer), and their properties affected airflow detection when subjected to different flow. The second test examined how different whisker array configurations affected similar flow information on oscillation and deflection.

A. Set up of experiments

A Stanley pivoting fan (ST-3130A-120, model) was used to simulate flow conditions at 3 different speeds of 300, 240 and 200 Cubic Feet per Minute (CFM) chronologically corresponding to High, Medium and Low (H, M, L) respectively. A set-up was created, creating a specific distance of exactly 9 inches and a flow angle of 36 degrees from the plane (Fig.2), parallel to the elastomer. Air flow velocity at the whiskers is not measured.

A setup was designed for the whisker placements. We used, 7.2mm, 3.6mm and 1.5mm thickness platinum-catalyzed silicon (Ecoflex) at weights of 4grams, 8grams and 2grams and a diameter of 64mm respectively to perform the material test.

The WhiskSight design secures and tensions the elastomer which holds a single magnetic attachment. The magnetic whisker is made using 1K, DH12 (K&J Magnetics) cylindrical magnets that holds the whisker to the elastomer (Fig.1). The whisker (100 mm rigid carbon fiber rod) is kept perpendicular to the elastomer surface and DH12 cylindrical magnets using a N42, NI, R212 (K & J Magnetics) ring magnet.

The elastomer was painted white using acrylic paint to prevent ambient light from penetrating. Red paints on the magnets indicate the sensor to a segmentation algorithm. Black dots made on the elastomer and magnet surface using permanent marker are tracked by a computer vision algorithm. Airflow is applied at H, M, L, and a video is recorded within a time range between zero to 15 seconds for each trial respectively. The video is then processed, and a single interface web camera is used as a transducer to track the interaction and sensor information with the computer vision algorithm.

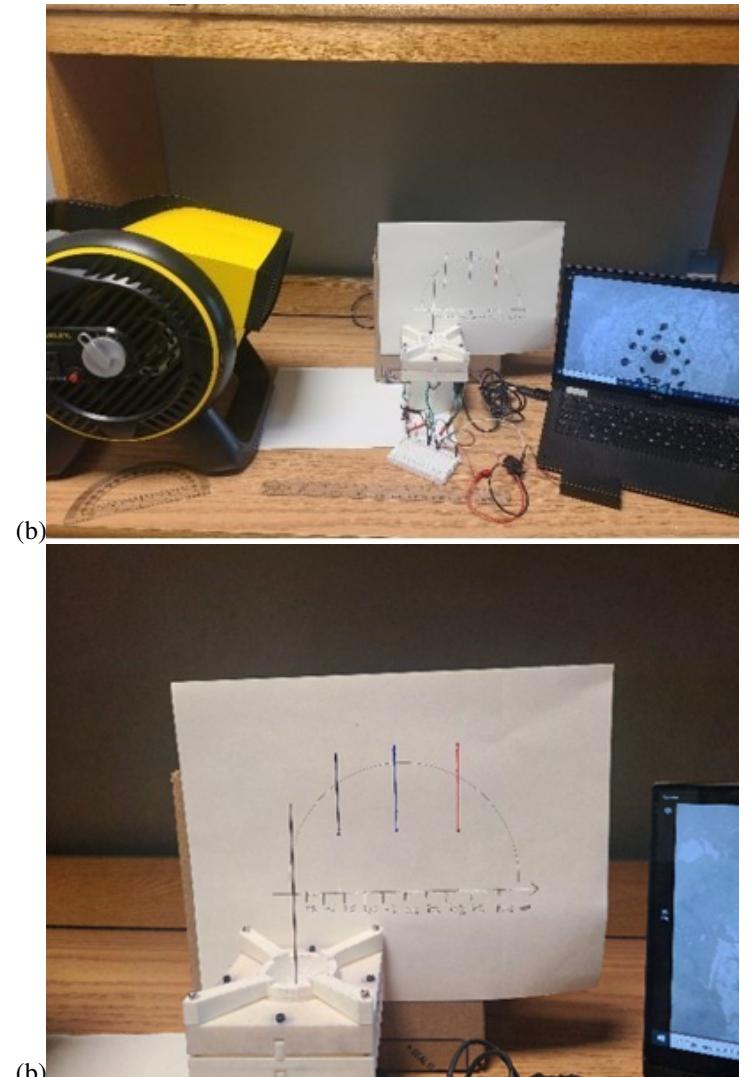


Fig. 2. Side view of airflow simulation for a single magnetic attachment on the WhiskSight sensor. Image on right, close view of calibrated workspace to physically observe whisker movement

B. Elastomer material substrate

Different materials and their properties can affect airflow detection when subjected to different flow. Using a platinum-catalyzed silicon plastic (Ecoflex), we studied the effects of airflow following the experimental procedures stated above. Consecutive tests were performed on 3.6mm 4grams and 7.2mm 8grams Ecoflex to study the whisker response to different material types. A following experiment examined how variations in the thickness of Ecoflex material affected the response of whiskers under the same airflow conditions. Here, we compared 3.6mm Ecoflex against 7.2mm Ecoflex at H, M, L respectively.

Using a thinner material, additional tests were conducted with 1.5mm 2grams Ecoflex material substrate, to verify the same hypotheses against already the existing experiments discussed above. This repetition was done to obtain additional and different trends of data and to further amplify our results.

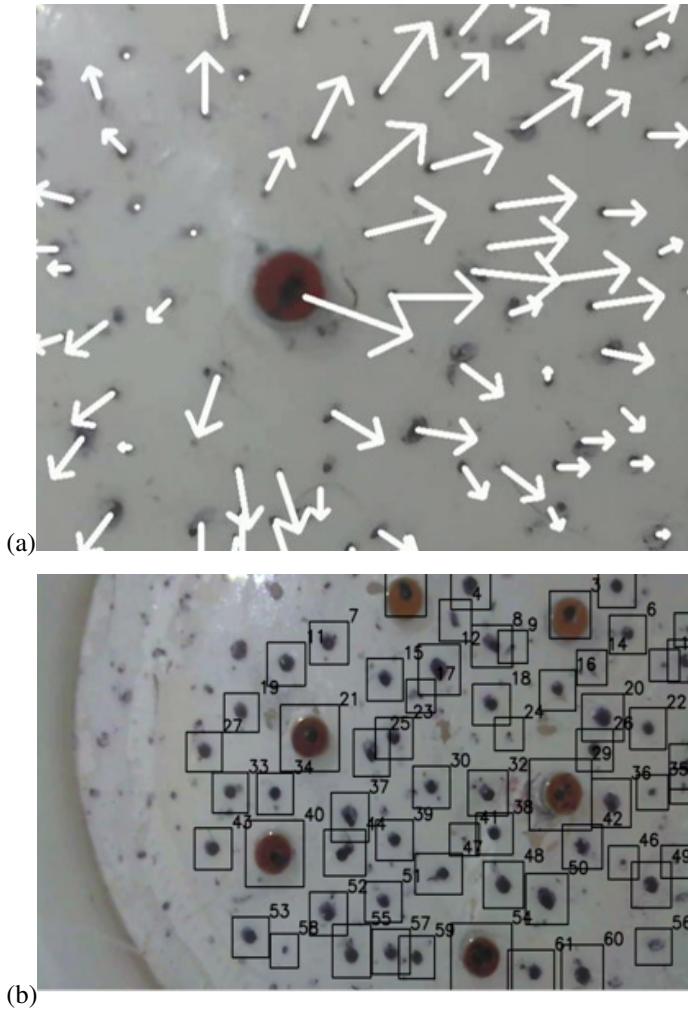


Fig. 3. (a) Translation of the dots from the initial frame are represented by an arrow which is 10x the translation in pixels. (b) Using a computer vision algorithm, all dots are identified by a segmentation algorithm are tracked.

C. Configuration of whisker arrays

A second experiment was conducted, where the whisker response to arrays with partial and no obstruction of flow based on whisker placements in the array. In this experiment, two magnetic rigid whiskers were mounted in a diagonal and perpendicular configuration (Fig.4). The direction of flow and whisker arrangement has been hypothesized to impact whisker output and behavior. The results from these experiments are compared to test if this is true.

III. TRACKING WHISKER MOTION AND FLOW OBSERVATION

We use a Python Minimum Output Sum of Squared Error (MOSSE) Tracker in a computer vision algorithm to track and interpret flow signals generated by the whisker. Each black dot in a square represents a potential whisker magnified 10 times the pixel translation of the dot by the tracking algorithm. The arrows represent whisker motion and point towards the direction of rotation and deflection of the whisker. (Fig.3,a)

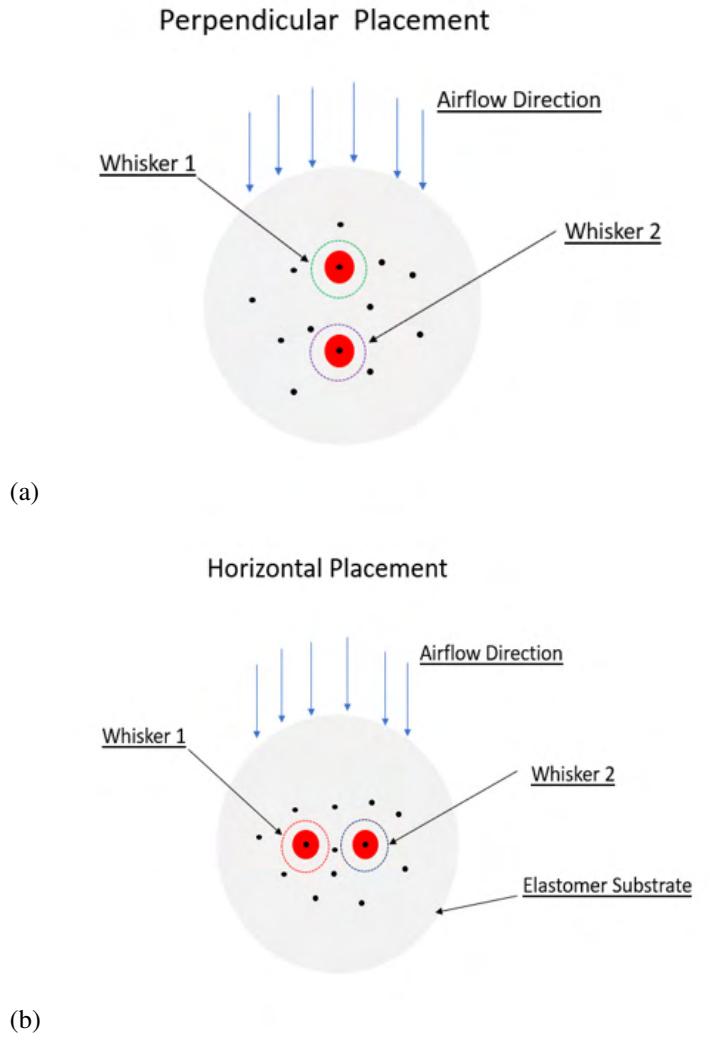


Fig. 4. (a) Ecoflex elastomer membrane, attached single magnet (red) and tracker points (black).Image (b) shows computer vision algorithm, tracking airflow direction on attached whisker on elastomer

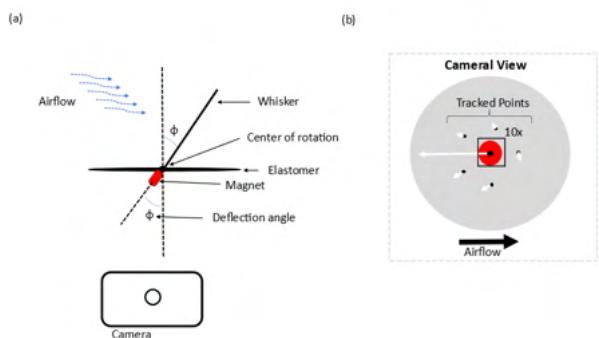


Fig. 5. (a)Schematic representation whisker deflecting angle when airflow is applied. (b) The direction of whisker movement from camera view (red magnet) is indicated by the computer vison and segmentation algorithm.

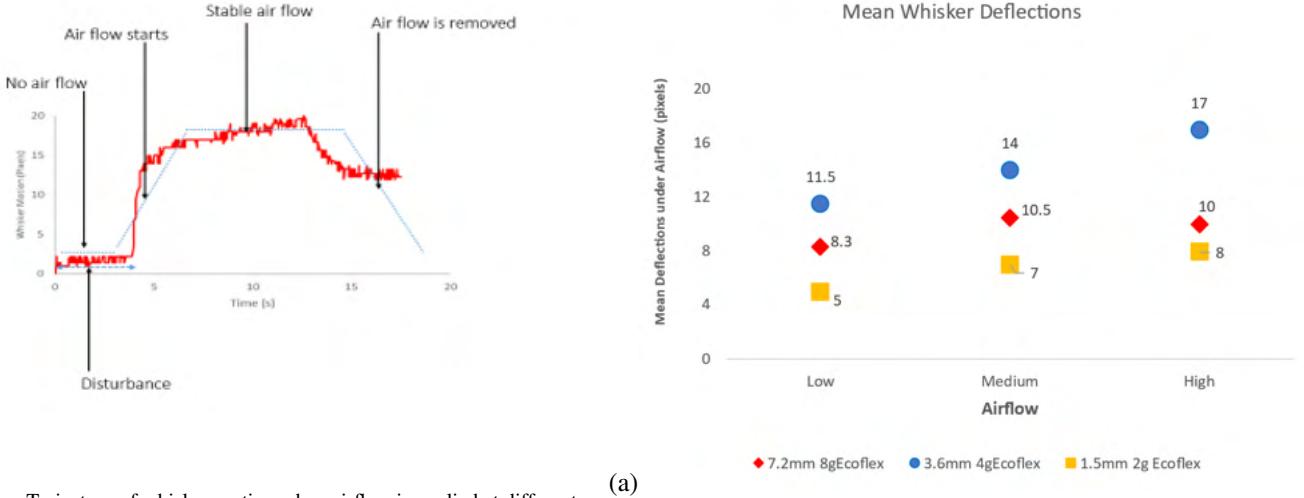


Fig. 6. Flow Trajectory of whisker motion when airflow is applied at different speeds and ranges

During each test, the setup and whisker test were recorded for each trial. This is important for close observation and performing checks to ensure the data obtained from the simulated results closely matches the whisker behavior during experiment. (Fig.2,b) Percentile bindings were calibrated on sheets of graphing and A4 white paper Boards to observe how far directly or physically the whisker deflects when a specific speed of airflow was applied. For example, we observed that, high air flow speeds caused the whisker to deflect more. This deflection was physically and analytically varied across different elastomer membranes to study their effects on the whisker

IV. RESULTS.

We find that, the mean whisker deflections produced directly correlate with the changes in airflow speed. This can be seen from the values of the distribution (Fig 7, a) which increase simultaneously from low to higher airflow. Additionally, it was also observed that smaller mean deflections values corresponded to points of lower material thickness. When the three different elastomer thicknesses were tested for affect, the 3.6mm elastomer (4grams) had the largest whisker deflection while the thinnest elastomer 1.5mm (2grams) had the smallest response.

This indication could mean that the thickness of the elastomer material could be a property of the deflections produced resulting from the whisker when under airflow motion. But the precise relationship between material thickness and sensor responsiveness is not clear although at each flow level the ranking of the whisker responsiveness remained unchanged.

Contrary to the expected observations from the average oscillations generated (Fig 7, b) there were no definitive correlations between the thickness of the material used (elastomer). This is known by looking at the plot for the average oscillations, which show the distribution order changing frequently. (Fig.7,b)

Because the material used had the same young's modulus coefficient of 200kpa, (Table 1, 2 3) the variation in deflection



Fig. 7. Average whisker oscillation and mean deflections for different elastomer thickness when different airflow speeds are applied

an oscillation values were thought to result from the difference in the thickness of the material as all other variables were kept constant. Tables oscillation and deflection values from whisker motions observed for 1.5mm, 3.6mm and 7.2mm Ecoflex respectively.

To directly observe the relationship between the different whiskers when airflow was applied, we calculated the change in deflection values (Δc) resulting from the different shapes of the whisker placements. (Table.2)

$$(PPD - HTL) = |\Delta c| \quad (1)$$

where PPD and HTL represent the perpendicular and horizontal shapes respectively. Changes in magnitude were observed for airflow distance between the two planes of the two whisker shapes resulting from the deflections, which increased simultaneously from low airflow to medium and high airflow for each of the resulting trials that were conducted. (Fig 8)

For the two varied orientations, the perpendicular configuration Generally received higher whisker deflections for each set of trials, with the first whisker for this configuration being

Table. 1 (a, b, c) oscillation and deflection values from whisker motions observed for 1.5mm, 3.6mm and 7.2mm Ecoflex respectively.

Flow rate	Mean Value	Oscillation	Thickness	Youngs modulus
High	17 Pixels	3.8/s	3.6mm	200 kpa
Medium	14 Pixels	3/s	3.6mm	200 kpa
Low	11.5 Pixels	4.3/s	3.6mm	200 kpa

(a)

Flow rate	Mean Value	Oscillation	Thickness	Youngs modulus
High	10 Pixels	5/s	7.2 mm	200 kpa
Medium	10.5 Pixels	4.8/s	7.2mm	200 kpa
Low	8.3 Pixels	4.3/s	7.2mm	200 kpa

(b)

Flow rate	Mean Value	Oscillation	Thickness	Youngs modulus
High	8 Pixels	5.8/s	1.5mm	200 kpa
Medium	7 Pixels	3.5/s	1.5mm	200 kpa
Low	5 Pixels	2.8/s	1.5mm	200 kpa

(c)

at the receiving front with higher mean deflection values. (Fig 7, Table 1,23). We expect to have a larger deflection angle for the first whisker in this placement and a smaller deflection angle for the second whisker due to the difference in airflow magnitude. This change in the amount of airflow on the whisker array also affected the drag profile for the second whiskers as the observations showed little movement towards its boundary separation (whisker2).

Observations from the horizontal whisker configurations revealed that airflow applied towards the whiskers was received at similar mean magnitudes of deflection. With Δc being

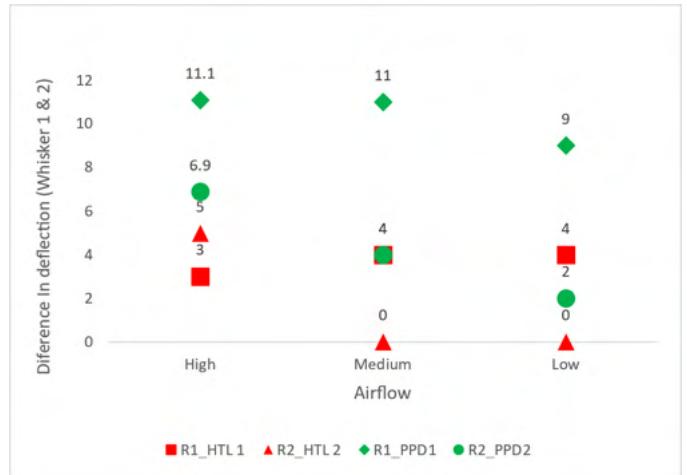


Fig. 8. Difference in whisker deflection for two trials at on perpendicular shape (PPD 1 2 and Horizontal Shape HTL 1 2

Table. 2 (a, b) Corresponding Δc values for first and second trial for a Horizontal and Perpendicular array placement

Airflow	Shape of Array		Shape of Array	
	R1_HTL 1	R1_PPD1	R2_HTL 1	R2_PPD1
High	3	11.1	5	6.9
Medium	4	11	0	4
Low	4	9	0	2

(a) (b)

equally similar at different airflows for the different trials. The amount of flow received by each of the whiskers for this configuration was observed to be within the same range along with movement towards the drag profile of the whiskers.

V. DISCUSSION AND FUTURE WORK

Frequency of oscillations and mean deflections of whisker may be indicators of flow speed and flow characteristics. The resulting number of whisker deflections produced, and the results are expected to show a closer and direct relationship between the deflecting angle and the magnitude of the resulting stimuli from the whisker. In this case flow, inertia drag and contact.

The placement of whiskers in different arrays or shapes affects the flow signals picked by the WhiskSight sensor when airflow is applied. We observe the relationship between a perpendicular (PPD) and a horizontal (HTL) placement and our results demonstrate that, the properties of each whisker vary relative to their position within the boundary or placement towards an airflow source. While eventually this may be a technique we can use to our benefit, at present it is only notable as a circumstance to be overcome in the design of the array.

For PPD configurations, each of the whisker within this set will have different deflections angles and changes towards the drag profile as airflow changes. For the horizontal configuration, the deflection angles will remain within a similar range for each of the whiskers as airflow is being applied.

Currently, the results from this study highlight material thickness as an important property, in tuning of flow on

Table. 3 First trial Horizontal Test (R1 HTL) Whisker 1 (a), Whisker 2 (b)

(a)			(b)		
Flow rate	Mean Value	Oscillation	Flow rate	Mean Value	Oscillation
High	15 pixels	8.4/second	High	12 pixels	7.4/second
Medium	12.5 pixels	3.6/second	Medium	8.5 pixels	7.6/second
Low	12 pixels	4.6/second	Low	8 pixels	7.8/second

Table. 4 Second trial Horizontal Test (R2 HTL) Whisker 1 (a), Whisker 2 (b)

(a)			(b)		
Flow rate	Mean Value	Oscillation	Flow rate	Mean Value	Oscillation
High	22 pixels	7.2 /second	High	17 pixels	6.4/second
Medium	19 pixels	4.4/second	Medium	19 pixels	6.6/second
Low	13 pixels	3/second	Low	13 pixels	3.6/second

Table. 5 First trial Perpendicular Test (R1 PPD) Whisker 1 (a), Whisker 2 (b)

(a)			(b)		
Flow rate	Mean Value	Oscillation	Flow rate	Mean Value	Oscillation
High	14 pixels	8 /second	High	7.1 pixels	7.8 /second
Medium	10 pixels	7.2 /second	Medium	6 pixels	6 /second
Low	12 pixels	6.6 /second	Low	10 pixels	8 /second

Table. 5 Second trial Perpendicular Test (R2 PPD) Whisker 1 (a), Whisker 2 (b)

(a)			(b)		
Flow rate	Mean Value	Oscillation	Flow rate	Mean Value	Oscillation
High	30.3 pixels	6.8/second	High	19.2 pixels	4.6/second
Medium	27 pixels	6.4/second	Medium	16 pixels	8.4/second
Low	22 pixels	7./second	Low	13 pixels	5.8/second

whiskers. Obviously, the information received from each whisker will independently vary based on their position or array configuration within their placement.

It is counter intuitive to these authors that the flow is maximized at a specific elastomer thickness. Further tests need to be performed in the future to verify this relationship. In the interim, it is certain that modifying the thickness of elastomer has a significant impact on the sensitivity of the whisker sensor.

These studied properties are important to help understand airflow and its effects on whiskers. If flow tuning is achieved, the integration of the WhiskSight sensor could find new applications in multidimensional sensing for robotics navigation in robust environment or terrain, flow detection in Unmanned Aerial vehicles and underwater systems such as submarines.

VI. ACKNOWLEDGMENT

This research was supported by the Carnegie Mellon University Robotics Institute and funded by the National Science Foundation (NSF).

The Author would like to thank Dr. Sarah Bergbreiter and Teressa Kent for instructions on handling experiments and evaluating results for this study.

VII. REFERENCES

[1] M. N. Muchlinski, J. R. Wible, I. Corfe, M. Sullivan, and R. A. Grant, “Good vibrations: The evolution of whisking in small mammals,” Anat. Rec., vol. 303, no. 1, pp. 89–99, 2020.

[2] Jie An, Pengfei Chen, Ziming Wang, Andy Berbille, Hao Pang, Yang Jiang, Tao Jiang, Zhong Lin Wang. Biomimetic Hairy Whiskers for Robotic Skin Tactility. 08 May 2021

[3] Martin J. Pearson1, Ben Mitchinson, J. Charles Sullivan, Anthony G. Pipe and Tony J. Prescott Biomimetic vibrissal sensing for robots . Bristol Robotics Laboratory, University of the West of England

[4] Andrea Tagliabue, Aleix Paris, Suhan Kim , Regan Kubicek , Sarah Bergbreiter , Jonathan P. How. Touch the Wind: Simultaneous Airflow, Drag and Interaction Sensing on a Multirotor

[5] J.B. Donnet, T.K. Wang, Carbon Fibers. Encyclopedia of Physical Science and Technology (Third Edition), 2003 <https://www.sciencedirect.com/topics/chemistry/youngs-modulus>: :text=1

[6] Teresa A Kent , Suhan Kim , Gabriel Kornilowicz , Wenzhen Yuan , Member, IEEE, Mitra J. Z. Hartmann and Sarah Bergbreiter , Member, IEEE. WhiskSight: A Reconfigurable, Vision-Based, Optical Whisker Sensing Array for Simultaneous Contact, Airflow, and Inertia Stimulus Detection

[7] Deepa L. Ramamurthy, Leah A. Krubitzer. The evolution of whisker-mediated somatosensation in mammals: sensory processing in barrelless S1 cortex of a marsupial, *Monodelphis domestica* <https://onlinelibrary.wiley.com/doi/10.1002/adma.202101891?af=R>

[8] By Bret Stetka. Seals Use Their Whiskers to See and Hear

<https://www.theatlantic.com/science/archive/2015/10/seals-use-their-whiskers-to-see-and-hear/413269/>

[9] Schulte-Pelkum, N Wieskotten, Sven Hanke, Wolf Dehnhardt, Guido Mauck, B. (2007). Tracking of biogenic hydrodynamic trails in harbor seals (*Phoca vitulina*). *The Journal of experimental biology.* 210. 781-7. 10.1242/jeb.02708.

Replicating Bugs Faster

Evolone Layne¹ and Jack Mostow²

Abstract—It is difficult for developers to replicate bugs based on a trial and error process. A bug can be classified as “stuck” or “crash”. This project focuses on developing a bug replication process for RoboTutor, an Android tablet application. A stuck bug occurs when a RoboTutor activity stays on the screen and can only be escaped by quitting the activity. A crash bug occurs when RoboTutor closes without being prompted to do so. This project explores a way to replicate and document bugs by creating a “bug recipe”. A bug recipe contains a screen recording of the visible behavior leading up to the bug, along with a timestamp, and other features that help developers diagnose bugs and debug RoboTutor. This project is important because this approach can serve as a blueprint that other developers can follow to speed up the bug replication process and debug their code.

Index Terms—Software Architecture for Robotic and Automation

I. INTRODUCTION

A. What is RoboTutor?

RoboTutor [1] is an application built to teach children in Tanzanian villages with little or no access to schools basic Swahili literacy and numeracy. RoboTutor was entered to compete in XPRIZE, a competition to develop software that give children the ability to teach themselves basic reading, writing and arithmetic within a 15 month period. While children in 28 villages in Tanzania used RoboTutor, gigabytes of detailed log files including thousands of instances of “crash” and “stuck” bugs that prevented the users from finishing activities were documented.

Before I joined the RoboTutor team, the developers were finding errors, but they found it laborious to replicate and produce the same bug without proper documentation. Bugs delay the progress made on applications, so instead of focusing on enhancing the features of the application, developers have to focus on making the application more functional. If less time were spent on the debugging process, applications could have more features.

My project aims to create and refine an efficient way to replicate bugs within software. This research is important because a better way to replicate bugs could speed up the debugging process. Previously, developers were replicating bugs by rerunning activities that sometimes led to them, and trying to remember the sequence of actions leading up to them. However, this is not an efficient method because replicating the exact error from memory after seeing it once

¹Evolone Layne is with the Department of Electrical Engineering and Computer Science, Howard University, Washington D.C., USA evolone.layne@bison.howard.edu

²Dr. Jack Mostow is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA mostow@cs.cmu.edu

can be extremely difficult. The objective of my research is to develop a way to replicate bugs faster.

II. METHODS

Before opening the application, I consulted the activity table (Table 1). The activity table consists of logged bugs and their probability of “crashing” or getting “stuck”. I used a pivot table to filter the large table and condense it to where I can see the stories with the highest stuck and crash rates. This made it easier to focus on replicating activities had high crash and stuck rates. A developer menu is a menu that only developers can see to select an activity. To bypass this part of the replication process, I had to create a custom story data.json file. This file consists of type, skill, tutor id, tutor desc, tutor data, and hash name.

Once the debug.json file is added, RoboTutor opens on the exact activity specified in the debug.json file (Figure 1). This allows for developers to go straight to the activity with a bug to start the replication process. In addition, a screen recording is taken every time an activity is initiated. The screen recordings are useful because they display the steps taken by the user to create a bug. This means that the same sequence of steps should cause the application to get stuck or crash. Now, the developers can isolate the error(s) and replicate the bugs faster.

Using this approach, I was able to obtain a screen recording of a stuck bug and follow the same actions shown in the recording several times to ensure that they constitute a reliable bug recipe. I hope to find more bugs using this documentation method.

```
{
  "type": "TRANSITION",
  "skill": "stories",
  "tutor_id": "story.hear::story_65",
  "tutor_desc": "story.hear",
  "tutor_data": "[unpackaged_asset]story_65",
  "use_hash_name": "false"
}
```

Fig. 1. This is the debug.json file used to launch the activity with the bug. Without this input, the developers would have to launch the app and manually go through the actions, which is time-consuming.

```
Activity ( times) : story.. hear-- story_ 65 (282)
Crash rate : 3.90%
% stuck 10s : 22.34%
Mean items started : 3.01
Mean items completed : 2.91
```

Fig. 2. These statistics are derived from the activity table. Bugs to replicate are chosen based on high stuck or crash rates.

A. Our Method

- 1) Bypass debugger menu.
- 2) Use custom story activity to skip to sentence.
- 3) Record video in case bug recurs.
- 4) If bug recurs, replay preceding steps as bug recipe.
- 5) If it doesn't, infer that bug was fixed.
- 6) Repeat to test reliability of recipe or bug fix.



Fig. 3. RoboTutor screenshot from an activity with a stuck bug.

III. DISCUSSION

A. Stuck Bug Example

One example of stuck bug occurred in story 33 (Figure 3). I selected this activity based on the fact that it had a high stuck rate. While playing the activity, the highlighted sentence was not read aloud. Instead of continuing with the story, it was stuck on this sentence, but the audio continued. This is an example of a stuck bug that we were able to replicate, thanks to the screen recording feature.

B. Sources of Delay

While doing this project, a few factors delayed progress. For example, a lot of the work being done relied on code that was not ready yet. In addition, analyzing the activity table is very difficult at the start of the project. This can slow one down at first, but after filtering this out and putting some of the data in a pivot table, it becomes much easier to handle.

Another source of delay came from the implementation of the debug.json file. When I first followed the format created prior to this project, I was stumped because whenever I ran RoboTutor, it would launch with a grey screen, which probably means that the unpackaged assets were not found.

One delay that is in fact a good delay is when we spend time on a bug that was fixed prior to the start of the project. The table reflects gigabytes of logged bugs, so keeping track of fixed bugs can be difficult. Creating a way to do this might also be beneficial in the future.

IV. EVALUATION

We will evaluate our research based on the number of bugs we are able to replicate. After this, we will assess the number of bugs successfully replicated. This criterion aids in the development of a percentage of total successful bug recipes. Another factor in the evaluation process is the amount of time it took to replicate the bug. This is because our goal is

to develop a faster way to replicate bugs in software. If our method takes less time, this means that we have achieved our goal of creating a faster way to replicate bugs. If bugs are not replicated throughout the process, we will evaluate the amount of recipe tests. This can also mean that the bug has been fixed prior to the start of this project.

V. CONCLUSION

Replicating bugs based on a trial and error process is tedious and can potentially produce even more bugs along the way. The focus of this project is to develop a more refined bug replication process for RoboTutor, an Android tablet application. The goal is to have the soon to be developed bug recipe process applied to other projects in this field. It will serve as a blueprint that other developers can follow to not only speed up the bug replication process, but to stay organized in the process.

VI. FUTURE WORK

In the future, we will try this method in practice. The goal is to continue to run RoboTutor and replicate frequently occurring bugs based on the activity table. Accumulating more bug recipes would help us to evaluate our method and prove that it is more reliable than the current replication process.

Extending our method to other activities is also very important. Right now, this method is being tested for use on story activities. Over time, getting this replication process to work for writing and math activities would be beneficial.

A. Continuation

To continue working on this method, it will be helpful to keep filter the table by highest stuck and crash rate because the table is very large. It may also be useful to get the debug.json working and past the grey screen to avoid the debugger menu.

ACKNOWLEDGMENTS

The authors would like to thank Rachel Burcin, Dr. John Dolan, the CMU Robotics Institute Summer Scholars (RISS) program organizers, and sponsors for making this program possible. The Global Learning XPRIZE created the opportunity for this project. Numerous volunteers, generous donors, and the \$1M Finalist Award supported the development of RoboTutor. The children who used RoboTutor generated the data. Mononito Goswami generated and refined the activity table.

REFERENCES

- [1] A. McReynolds, S. Naderzad, M. Goswami, and J. Mostow (2020). Toward Learning at Scale in Developing Countries: Lessons from the Global Learning XPRIZE Field Study. In J. Peters, Ed., L@S '20: Proceedings of the Seventh ACM Conference on Learning @ Scale. Association for Computing Machinery: Virtual Event, USA, August 12-14, 2020, vol. 3, pp. 175—183. McGraw-Hill.

Multi-agent Hierarchical Reinforcement Learning in Urban and Search Rescue

Long Le¹ and Dana Hughes² and Katia Sycara²

Abstract—In an urban search-and-rescue (USAR) task, a team of agents cooperates to explore different rooms in the environment, clear rubble, and triage victims. USAR presents a challenging control problem in multi-agent Reinforcement Learning due to its long horizon, sparse and delayed reward, and the large size of the state space. Hierarchical approaches such as options and state abstraction have been proposed as a way to reduce the state space and planning complexity in such problems. In this work, we leverage domain knowledge to (1) train individual low-level (sub-)policies of each agent in smaller subsets of the state space, and (2) then train a team-level policy over a reduced graph representation of the states using those pretrained sub-policies. We show some early-stage results where our approach outperforms two multi-agent algorithms option-critic, and independent Q-learning on a simple environment.

Index Terms—Multi-agent Reinforcement Learning. Hierarchical Planning. Search-and-Rescue.

I. INTRODUCTION

Reinforcement Learning (RL) is a framework where an agent or multiple agents interact with the environment, receive observations and rewards for their actions. The agents learn through experience how to maximize their future discounted rewards. RL has been applied to solve a diverse array of domains, from game playing [Mnih et al., 2013] to self-driving [Kiran et al., 2021], and provides a natural way for training artificial teams in urban search-and-rescue tasks.

In a USAR task, a fully cooperative team of agents has to navigate and explore the environment, removing obstacles such as rubble along the way, to locate and rescue victims. This is a difficult task in several ways. First, USAR is a *hard-exploration* problem where the environment has very sparse and delayed rewards. That is, because the objective of the team is to rescue victims, agents can only receive extrinsic rewards when a victim is successfully triaged, which is a rare event. Further, the learning is complicated by *delayed* rewards [Arjona-Medina et al., 2018] in that the decision an agent makes in one time step tends to have its consequence revealed much later. For example, when the agent clears the rubble at the entrance of a large hall-way, it might only know if its effort has been worth it when a teammate discovers a victim in the hall-way multiple time steps later. The learning can also suffer from *long horizon*. Coupled with sparse rewards, long horizon means that an agent needs to perform a very long sequence of actions before any reward can be received. Finally, environments

Hierarchical Model: Overall

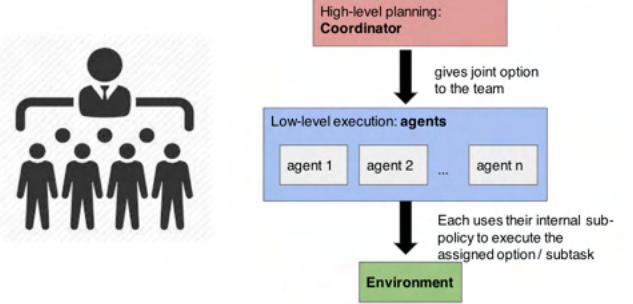


Fig. 1: Coordinator-Agent Hierarchical Framework.

for USAR missions are typically large, requiring agents to differentiate and generalize between a vast number of states.

To deal with hard-exploration and large state space, hierarchical approaches, both over the action space [Dayan and Hinton, 1992], [Sutton et al., 1999] and the state space [Barry et al., 2011], [Steccanella et al., 2021], [Shang et al., 2019], have been proposed. In this work, we utilize both action and state hierarchical representations to tackle the USAR problem. Specifically, we train each agent in the team to complete some sub-tasks in a completely decentralized manner using the low-level features of subsets of the state space, and then train a centralized team-level coordinator on a high-level graph representation of the environment. In our framework, the coordinator gives command to the team, for example “agent 1 goes to search for victims in room A, and agent 2 goes to room B”. The agents execute their assigned sub-tasks and report their rewards to the coordinator. We do not assume an agent’s sub-policy to achieve a sub-task is prescribed a priori but instead choose to obtain it through decentralized training. See Fig. 1 for an illustration. This approach reduces the state space since an agent only plans over a small subset of the environment, and the coordinator only plans over a small graph view. It also alleviates the long planning horizon problem since the coordinator avoids micro-management and learns to associate high-level decisions with rewards. We demonstrate that this approach outperforms two multi-agent algorithms option-critic, and independent Q-learning in a simple environment.

Training high-performing agents not only can contribute towards the future development of autonomous robotic teams for USAR missions but can also serve to give advice to human teams in order to improve their performance.

¹Long Le is with the College of Information and Computer Sciences, University of Massachusetts Amherst. lnle@umass.edu

²Sycara and Hughes are with the Robotics Institute, Carnegie Mellon University. {danahugh, sycara}@andrew.cmu.edu

The organization of this paper is as follows. Section II reviews single-agent and multi-agent Reinforcement Learning. Section III discusses some previous work to obtain hierarchical structures for action and state spaces. Our approach in modeling the search-and-rescue task using hierarchical planning is given in Section IV. The result is given in Section V. Section VII is the conclusion and future work.

II. PRELIMINARIES

A. Markov Decision Process

In single-agent RL, a Markov Decision Process (MDP) is often assumed. MDP is a tuple $(\mathcal{S}, \mathcal{A}, p_0, p, r, \gamma, L)$, where \mathcal{S} is a set of states ($s \in \mathcal{S}$), \mathcal{A} the action space ($a \in \mathcal{A}$), $p_0(s_0)$ the initial state distribution, $p(s_{t+1}|s_t, a_t)$ the environment dynamics, and $r(s_t, a_t, s_{t+1})$ the reward function, γ the discount factor, and $L \in \mathbb{N} \cup \{\infty\}$ the horizon. In words, the agent appears initially at state s_0 drawn from p_0 . At any time step t , the agent is in state s_t , takes an action a_t and transition to the next state s_{t+1} drawn from $p(\cdot|s_t, a_t)$, and receives a discounted reward of $\gamma^t r(s_t, a_t, s_{t+1})$. In finite-horizon problems such as ours, the agent keeps interacting with the environment until the time step reaches the horizon L , after which the episode is terminated. A policy $\pi(a_t|s_t)$ is a conditional probability distribution over actions, dictating which action the agent will take in a given state. The goal of the agent in RL is to learn a policy π that maximizes the expected total discounted reward

$$\mathbb{E} \left[\sum_{t=0}^L \gamma^t r(s_t, a_t) \mid s_0, \pi \right].$$

B. Partially Observable Markov Decision Process

A partially observable decision process (POMDP) is an extension of MDP to situations where the agent does not have full access to the environment state s but rather to only observation $o \in \mathcal{O}$ drawn from the observation probability model $O(o \mid s)$. In a USAR mission, an agent does not have perfect information about the environment, and can only access the observations. For example, the agent does not know where the victims are initially but can observe its local surroundings.

C. Decentralized Partially Observable Markov Decision Process

A Decentralized Partially Observable Markov Decision Process (Dec-POMDP) is a generalization of POMDP to multi-agent systems. Dec-POMDP allows for heterogeneous teams of agents by enabling agents to have different action sets and observations. Formally, an agent i can take action $a^i \in \mathcal{A}_i$ and receive observation $o^i \in \mathcal{O}^i$. The state transition probabilities $p(s'|s, a^1, \dots, a^n)$ and rewards $r(s, a^1, \dots, a^n, s')$, where n is the number of agents, depend on the joint action (a^1, \dots, a^n) of all agents. Note that the reward r in this case is shared among all agents. This is appropriate for fully-cooperative team settings such as ours. Markov games [Littman, 1994] extend Dec-POMDP to mixed cooperative-competitive settings by allowing different

agents to have different rewards. We will assume a Dec-POMDP in our work.

III. RELATED WORK

A. Action Abstraction

Two popular frameworks for hierarchical RL over the action space are options [Sutton et al., 1999] and Feudal learning [Dayan and Hinton, 1992], [Vezhnevets et al., 2017].

In the former approach, the concept of “macro-actions” or options is introduced. An option is a high-level action that consists of primitive actions and is *temporally extended* i.e. the option might take a variable number of time steps to complete. For example, the option of “going to the airport” consists of a sequence of primitives such as steering the car wheel, and might take 30 or 45 minutes depending on the traffic condition. Formally, an option ω consists of an initiation set $\mathcal{I}_\omega \subset \mathcal{S}$, a sub-policy $\pi_\omega(a|s)$ and a termination probability distribution $\beta_\omega(s)$. In state s , the agent uses a meta-control policy $\mu(\omega|s)$ to pick an option ω available, execute the option using π_ω until completion determined by β_ω . There are two levels to be learned μ and $\{\pi_{\omega_i}\}_{i=1}^m$ where m is the number of options. Options can be explicitly defined and learned using subgoals and pseudo-rewards as in [Sutton et al., 1999], or both levels can be learned simultaneously from end-to-end using option-critic in [Bacon et al., 2016]. Distributed option-critic [Chakravorty et al., 2019] extends the option-critic framework to multi-agent systems via common-belief and communication.

In the latter approach of Feudal learning, there is a hierarchy of managers (lords) and sub-managers (serfs). The managers at different levels in the hierarchy observe different scales of resolution of the environment, assign tasks to sub-managers or workers one level below them. A manager at a level receives an intrinsic reward from their superior based on how well they execute their assigned task. This framework is different from the option’s in that it focuses on state space abstraction between different levels of hierarchy rather than temporal abstraction [Riemer et al., 2019].

Our approach is inspired by both of the mentioned frameworks. We use sub-goals to train agents’ sub-policies, and use a coarser resolution of time and space for the coordinator (manager) for the team-level policy.

B. State Abstraction

In some domains, it is possible to reduce the state space by “factoring” the MDP [Boutilier et al., 2000]. The idea is that some parts of the MDP state are independent and thus can be exploited by a more compact Bayesian graphical representation. [Barry et al., 2011] presents an algorithm to cluster states of a factored MDP into “macro-states”. In the present work, we have not attempted to represent the states in factored form.

The authors in [Stecanella et al., 2021] learn a hierarchical representation of the environment by partitioning the states into a set of “abstract states”. A “compress function”, mapping primitive states to abstract states, is learned from data. The idea is to first collect a set of trajectories by letting

the agent wander through the environment. Then, a compress function is optimized using the heuristic that consecutive states in a trajectory should usually belong to the same abstract state¹. They showed that the learned abstract states correspond to different rooms in a grid-world experiment. Our current work assumes that the abstract states are already given.

The paper [Shang et al., 2019] is another approach to obtain a high-level view of the environment by identifying the “pivotal states”, where pivotal states are defined to be those most useful to predict the agent’s actions. A variational autoencoder is used to learn the pivotal states from collected trajectories. They show that the pivotal states correspond to hall-way junctions in 2-D mazes. This is an interesting direction for our future work.

Note that in both [Steccanella et al., 2021] and [Shang et al., 2019] approaches, we need to run some exploration trajectories to build a state abstraction before actual learning can occur. In this paper, we take a shortcut and use a pre-specified state abstraction instead.

IV. APPROACH

A. Two-rooms Environment

The two-rooms environment consists of a 10x15 2-D grid with two victims in the top and bottom rooms, and two medics in a hall-way (see Figure 2). The goal of the agents is to rescue both victims in the shortest amount of time possible. Each agent location is a tuple (x, y, dir) , where (x, y) is a coordinate on the 2-D grid and dir is one of four possible directions NORTH, SOUTH, WEST, EAST, representing the direction that the agent is currently facing. The navigation actions include FORWARD, TURN-LEFT and TURN-RIGHT. An agent also has a special action TRIAGE that only takes effect if the agent is facing a victim; otherwise, it is a no-op. The environment also has walls that the agents cannot pass through. In other words, a FORWARD action when facing a wall does nothing.

For simplicity, we assume a fully observable MDP. The horizon is set to 100, and γ to 0.99.

B. Graph-level View and Sub-policies Training

We handcraft a graph representation of the environment where each node is a section of the map (see Figure 3). The agents learn to navigate between neighboring nodes of the graph through pseudo-rewards. For example, to train a navigation strategy for the bottom agent from its position to the top room, we remove the other agents and victims from the map. Then, we give a synthetic reward of +100 for reaching any locations in the top room, and a reward of -1 with every passing time step to encourage the agent to reach the sub-goal as fast as possible. Within each room, we also train the agents to triage the victims by giving a reward of +100 for a successful triage and -1 time reward as before. Note that this triage sub-policy is learned assuming that an

¹One naive abstract function is to map all states to the same abstract state. [Steccanella et al., 2021] avoids this by having another loss term to make sure that all abstract states are somewhat equally likely over the trajectories.

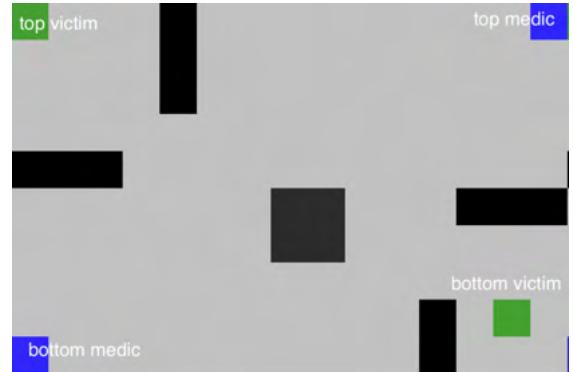


Fig. 2: The two-rooms environment. The walls are in black. Two green squares are two victims. Two blue squares are two agents.

agent is already in the room where a victim might be located. Thus, the sub-policy has no guarantee for good performance when, for example, an agent is asked to execute the option “RESCUE TOP VICTIM” when the agent is not in the top room but rather is in the hall-way.

Note that the agents’ sub-policies are local in the sense that they require local information for execution. For example, to execute the option of going from the hall-way to the top-room, an agent needs not know what is happening in the bottom room. Note that in our current work, since we assume a MDP, this discussion about local information is less relevant. However, in a Dec-POMDP version of the problem, sub-policies dependency on only local information would reduce the state representation needed for learning those policies.

Further, the approach of local sub-policies provides an implicit curriculum training [Bengio et al., 2009] for the agents, lessening the problem of sparse rewards and long horizon. For instance, when an agent is already in the top-room, it is much more likely that the agent stumbles upon triaging the top victim, thereby providing more frequent rewards in the initial exploration phase of an RL algorithm, and thus speed up learning.

The description of options is given in Table I. In general, we should train 4 navigation sub-policies: TOP-to-HALLWAY, HALLWAY-to-TOP, HALLWAY-to-BOTTOM, and BOTTOM-to-HALLWAY. Then, to navigate from the bottom room to the top room, an agent would chain together its BOTTOM-to-HALLWAY and HALLWAY-to-TOP options. For this simple environment, we take a shortcut and only train 2 navigation sub-policies as in Table I with the understanding that the agents are always initialized in the hall-way during the full search-and-rescue mission. Also, there is no need to jump 2 hops in the graph here (e.g. going from the bottom to the top room) since an optimal team-level policy is to have the top medic go to the bottom room, and the bottom medic to the top room.

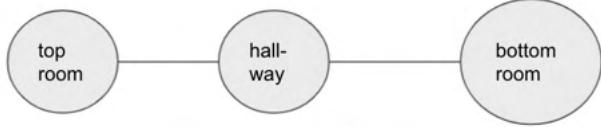


Fig. 3: The graph representation for the two-rooms environment.

C. Coordinator’s Policy Training

Once we have had the trained subpolicies, we proceed to train the coordinator’s policy over the joint option using the graph representation. That is, the coordinator sees a state (n_1, n_2, v_1, v_2) where $n_i \in \{\text{TOP-ROOM}, \text{HALL-WAY}, \text{BOTTOM-ROOM}\}$ indicates the position of agent i , and $v_i \in \{0, 1\}$ indicates whether the i^{th} victim has been triaged. The coordinator’s meta-policy is over the joint options (ω_1, ω_2) where each ω_i can take on 1 of 4 possible options (see Table I).

The coordinator selects a joint option and commands its agents to use their sub-policies to execute that joint option. Once one of the agents completes its option, the coordinator terminates the options of all agents, re-evaluates the situation, and select another joint option. We also enforce the termination conditions of options by giving a large negative reward to the coordinator whenever it tries to select an unavailable option. For example, if the top agent is in the hall-way and the coordinator commands it to execute “TRIAGE-BOTTOM-VICTIM”, the coordinator will receive a large negative reward and learn to avoid selecting invalid options in the future.

V. RESULT

Both the training of sub-policies and the coordinator’s meta-policy are done by Deep Q networks with experience replay and target nets [Mnih et al., 2013].

We compare our result against two multi-agent algorithms: the option-critic [Sutton et al., 1999], and independent Q-learning.

For each algorithm, we run 8 trials and plot the results in Figures 4, 5, 6. Besides rewards, we also plot the number of time steps taken in each episode. The lower the number of steps, the quicker the agents were able to achieve their goals. If the agent was not able to accomplish its goal, then the number of steps would be equal to the horizon. For every small number of training episodes, we also performed a test episode. DQN-based models sometimes suffer from “catastrophic forgetting” [Roderick et al., 2017] when the agent’s performance degrades after achieving optimal behavior. One reason is that after achieving a decent policy, the agent is only used to seeing “good” states. When some “bad” states are encountered by chance, the agent’s Q-value prediction is completely off, incurring a high error and changing the network weights unfavorably. The simple solution in [Mnih et al., 2013] is to regularly test the model during training

and save the model resulting in the best test performance so far.

Each of our sub-policy takes at most around 600 episodes to obtain optimal solutions. In the best trial, they take around 1000 episodes in total but they can be trained concurrently. The coordinator’s meta-policy trained on those sub-policies takes about 30 episodes to reach optimality. The best trial of the option-critic takes around 1,500 episodes to reach optimality. For independent Q-learning, the variance in performance is small but the algorithm was not able to rescue both victims at all. The best trial can learn to triage one victim after 450 episodes. The performance of various algorithms is summarized in Table II. We compare the best performance of each algorithm against each other since the average metric might be deceptive. As discussed, there is the problem of catastrophic forgetting so if there are two trials that achieve optimality at different times, for example the test rewards of the two trials are [0, 100] and [100, 0], then their average might seem to suggest that the agent has not achieved the goal at all: the average is [50, 50].

The independent Q-learning does not touch on the multi-agent aspect of the problem since each agent learns on its own, and is not aware of the existence of other agents. Thus, this approach suffers from non-stationarity as other agents change their policies over time.

The distributed option-critic overcomes the non-stationarity issue by giving the centralized critic access to all observations of agents. It also improves the MADDPG algorithm [Lowe et al., 2017] by adding action abstraction to this approach by using options. However, there is no state abstraction, and the options are learned, which might not be as good as hand-specified options using domain knowledge.

VI. CONCLUSION AND FUTURE WORK

In this work, we adopt hierarchical abstraction over the state and action spaces to build a team of autonomous agents for the urban search-and-rescue task. We present our preliminary results for a simple two-rooms environment, and compare them to those by common end-to-end multi-agent algorithms.

In the future, we would like to extend this framework to a more complicated environment of bigger size, with heterogeneous agents with different capabilities, rubble, and partial observability. For a bigger environment, we plan to use a Graph Neural Network (GNN) to encode the environment information and formulate the Reinforcement Learning problem as a node-level decision-making task (e.g. see [Gammelli et al., 2021]). In the case of partial observability, there is a problem: the agents do not know whether a room contains a victim, and if so how many and where they are. Here, intrinsic curiosity-driven exploration [Pathak et al., 2017] and multi-object search [Wandzel et al., 2019] can help. Also, we would like to automate the learning as much as possible, for example by learning the state representation through initial exploration as proposed by [Steccanella et al., 2021] and [Shang et al., 2019]. We would also like to experiment with transfer learning as discussed in [Steccanella

Option	Initiation set	Termination condition
NAVIGATE-TO-TOP-ROOM	agent is in the hall-way	agent reaches the top room
NAVIGATE-TO-TOP-ROOM	agent is in the hall-way	agent reaches the bottom room
TRIAGE-TOP-VICTIM	agent is in the top room	top victim is triaged
TRIAGE-BOTTOM-VICTIM	agent is in the bottom room	bottom victim is triaged

TABLE I: The agent’s options in the two-rooms environment.

Algorithms	Best number of eps til optimality in 8 trials
Graph + options (ours)	1030
Option-critic	1,500
Independent Q-learning	> 5,000

TABLE II: The performances of different multi-agent control algorithms.

et al., 2021] by randomizing the victim locations. Intermittent communication where the coordinator only has access to a “common belief” [Chakravorty et al., 2019] updated by broadcasting between agents is also possible.

Object-oriented model-based Reinforcement Learning [Diuk et al., 2008], [Wandzel et al., 2019] is also an attractive alternative to model-free RL like DQN we are currently using. Model-based RL is generally more sample-efficient, more generalizable and transferable than model-free RL when domain knowledge is available. In object-oriented RL, the environment consist of objects of different classes, and the experience with one object can be generalized to that with another object of the same class. For example, an object can be a wall in our case. The agent might learn that it cannot pass through a wall in Room 1. This experience can also be referenced when the agent encounters another wall in Room 2 under the model-based framework. In model-free, on the other hand, the agent does not have the concept of a wall. What it learns is that I cannot perform a certain move at a specific location in Room 1. When encountering the object of the same class in Room 2, it has to re-learn the fact that it cannot move through wall again.

Lastly, we would like to consider agent advising [Torrey and Taylor, 2013] and hierarchical active imitation learning [Le et al., 2018]. Under these frameworks, a teacher, which is a trained high-performant team of agents, would observe a student to provide feedback in order to accelerate the student’s learning. Under the agent advising paradigm, the teacher predicts the student’s next action, and intervene whenever it believes that the student is going to make a mistake. Under the hierarchical imitation approach, the teacher observes the student’s mistake after the fact and provides feedback, still in an online manner.

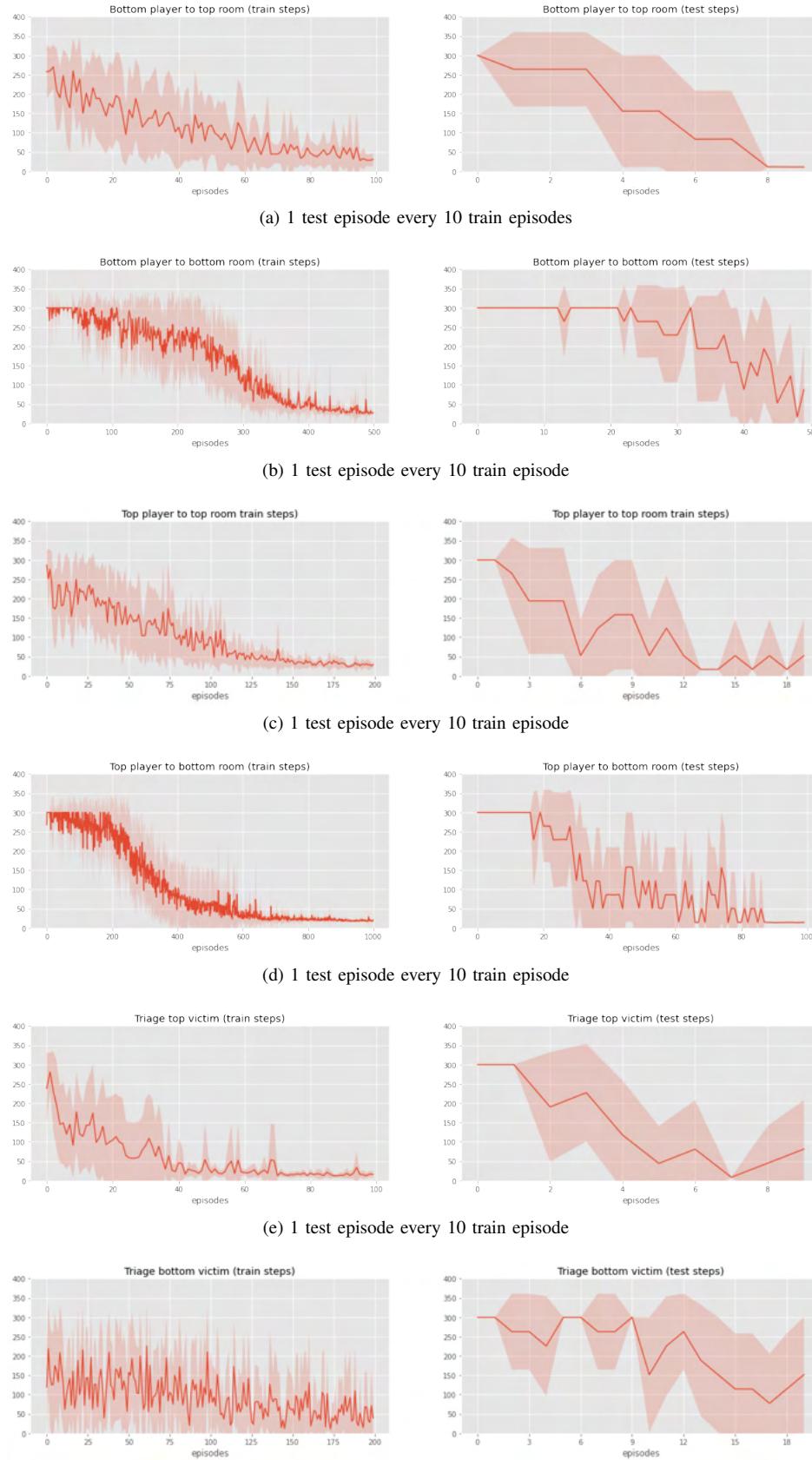
VII. ACKNOWLEDGEMENT

The author (Long Le) would like to thank Rachel Burcin, John Dolan, Jennie Piotrkowski and all members of the RISS cohort for making RISS a wonderful experience. The author would like to thank their mentors Katia Sycara and Dana Hughes. The author also benefited from the insights in multi-agent and hierarchical RL from Tejas Gupta and Sophie Guo.

REFERENCES

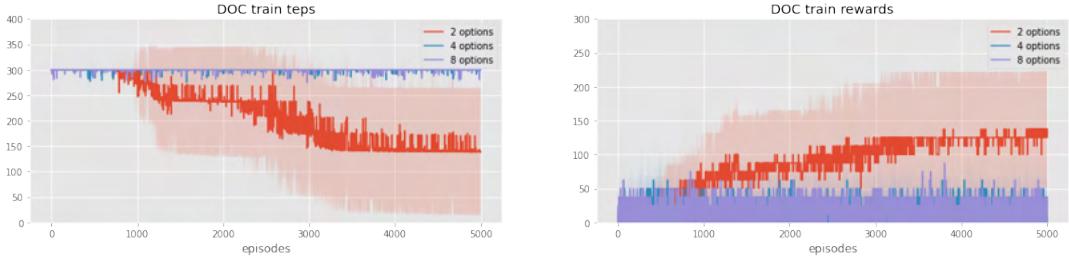
- [Arjona-Medina et al., 2018] Arjona-Medina, J. A., Gillhofer, M., Widrich, M., Unterthiner, T., and Hochreiter, S. (2018). RUDDER: return decomposition for delayed rewards. *CoRR*, abs/1806.07857.
- [Bacon et al., 2016] Bacon, P.-L., Harb, J., and Precup, D. (2016). The option-critic architecture.
- [Barry et al., 2011] Barry, J. L., Kaelbling, L., and Lozano-Perez, T. (2011). Deth*: Approximate hierarchical solution of large markov decision processes. In *IJCAI*.
- [Bengio et al., 2009] Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML ’09, page 41–48, New York, NY, USA. Association for Computing Machinery.
- [Boutilier et al., 2000] Boutilier, C., Dearden, R., and Goldszmidt, M. (2000). Stochastic dynamic programming with factored representations. *Artificial Intelligence*, 121(1):49–107.
- [Chakravorty et al., 2019] Chakravorty, J., Ward, P. N., Roy, J., Chevalier-Boisvert, M., Basu, S., Lupu, A., and Precup, D. (2019). Option-critic in cooperative multi-agent systems. *CoRR*, abs/1911.12825.
- [Dayan and Hinton, 1992] Dayan, P. and Hinton, G. E. (1992). Feudal reinforcement learning. In *Advances in Neural Information Processing Systems 5, [NIPS Conference]*, page 271–278, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- [Diuk et al., 2008] Diuk, C., Cohen, A., and Littman, M. L. (2008). An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML ’08, page 240–247, New York, NY, USA. Association for Computing Machinery.
- [Gammelli et al., 2021] Gammelli, D., Yang, K., Harrison, J., Rodrigues, F., Pereira, F. C., and Pavone, M. (2021). Graph neural network reinforcement learning for autonomous mobility-on-demand systems.
- [Kiran et al., 2021] Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A. A. A., Yogamani, S., and Pérez, P. (2021). Deep reinforcement learning for autonomous driving: A survey.
- [Le et al., 2018] Le, H. M., Jiang, N., Agarwal, A., Dudík, M., Yue, Y., and III, H. D. (2018). Hierarchical imitation and reinforcement learning. *CoRR*, abs/1803.00590.
- [Littman, 1994] Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *In Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163. Morgan Kaufmann.
- [Lowe et al., 2017] Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. *CoRR*, abs/1706.02275.
- [Mnih et al., 2013] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing atari with deep reinforcement learning.
- [Oroojlooyjadid and Hajinezhad, 2019] Oroojlooyjadid, A. and Hajinezhad, D. (2019). A review of cooperative multi-agent deep reinforcement learning. *CoRR*, abs/1908.03963.
- [Pathak et al., 2017] Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. *CoRR*, abs/1705.05363.

- [Riemer et al., 2019] Riemer, M., Liu, M., and Tesauro, G. (2019). Learning abstract options.
- [Roderick et al., 2017] Roderick, M., MacGlashan, J., and Tellex, S. (2017). Implementing the deep q-network.
- [Shang et al., 2019] Shang, W., Trott, A., Zheng, S., Xiong, C., and Socher, R. (2019). Learning world graphs to accelerate hierarchical reinforcement learning. *CoRR*, abs/1907.00664.
- [Steccanella et al., 2021] Steccanella, L., Totaro, S., and Jonsson, A. (2021). Hierarchical representation learning for markov decision processes.
- [Sutton et al., 1999] Sutton, R. S., Precup, D., and Singh, S. (1999). Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artif. Intell.*, 112(1–2):181–211.
- [Torrey and Taylor, 2013] Torrey, L. and Taylor, M. (2013). Teaching on a budget: Agents advising agents in reinforcement learning. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS ’13, page 1053–1060, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.
- [Vezhnevets et al., 2017] Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., and Kavukcuoglu, K. (2017). Feudal networks for hierarchical reinforcement learning. *CoRR*, abs/1703.01161.
- [Wandzel et al., 2019] Wandzel, A., Oh, Y., Fishman, M., Kumar, N., Wong, L. L., and Tellex, S. (2019). Multi-object search using object-oriented pomdps. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7194–7200.

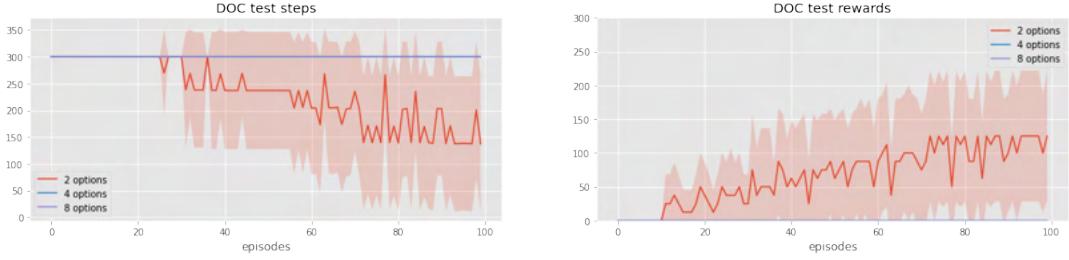


(f) 1 test episode every 10 train episode. For this sub-policy to succeed consistently across trials, we had to lower γ to 0.9. Since the bottom room is very small, having small γ prevented the agent from wandering to other parts of the environment. The usual $\gamma = 0.99$ still worked for one trial. In practice, it would not be necessary to adjust γ since we would save and use the best model in terms of test performance.

Fig. 4: Training sub-policies.

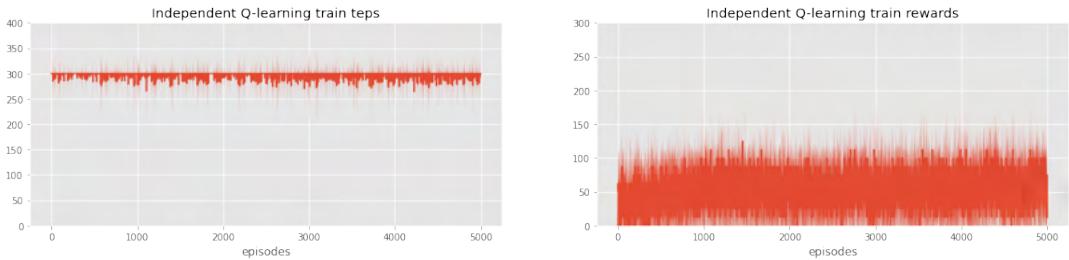


(a) Training performance of DOC

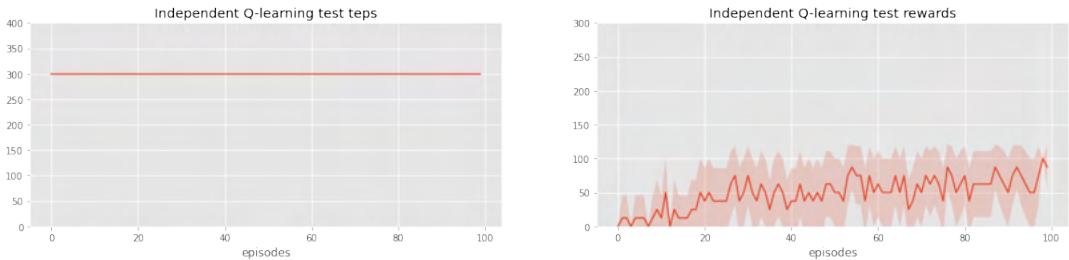


(b) Testing performance of DOC. For every 50 training episodes, we perform one test episode using a greedy-Q policy.

Fig. 5: Performance of the simplified distributed option-critic (DOC) algorithm. The number of options is a hyper-parameter in DOC. We set it to 2,4 and 8 since our approach uses 4 options.



(a) Training performance of Independent Q-learning.



(b) Testing performance of Independent Q-learning. For every 50 training episodes, we perform one test episode using a greedy-Q policy.

Fig. 6: Performance of independent Q-learning.

SiamFIND: Towards One-shot Informed InterestiNg Object Detection

Bowen Li¹, Chen Wang², and Sebastian Scherer²

Abstract—Recent years have witnessed the fast development and promising prospects of visual object detection methods. However, for robots’ autonomous exploration tasks, the method is expected to detect unseen interesting novel objects with online defined few labeled samples, where a few-shot learner is expected. Despite that prior works have made progress in few-shot object detection (FSOD) tasks, the number, as well as the members of the novel classes, are quite fixed after fine-tuning. There exist very few detectors suitable for robots’ online exploration, where the interesting objects are actually dynamic. To this end, this work proposes a novel Siamese Few-shot InterestiNg object Detector (SiamFIND), which builds favorable marriage between Faster RCNN and meta-learning strategy. Specifically, a Siamese branch is designed to extract intra-class prototypes, which enables a prototype embedding network for localization and a learnable cosine module for classification in the meanwhile. Once trained, the proposed SiamFIND can thus fast generalize to and accurately detect user-defined dynamic few-shot interesting objects without further updating. Exhaustive experiments have been conducted on the MSCOCO dataset, which demonstrated the superiority of SiamFIND against a baseline method. Furthermore, SiamFIND will be implemented in some image sequences captured during real-world exploration, strongly validating the feasibility of SiamFIND in such a task.

Index Terms—Object detection, few-shot learning, robot autonomous exploration

I. INTRODUCTION

Pre-trained on large-scale image datasets, object detectors aim at localizing and classifying the objects in a test image. Such algorithm has facilitated numerous applications for robots including grasping [1], navigation [2], and localization [3]. While for robots’ online autonomous exploration task, since the operation area is usually unknown, the objects are probably unseen in the training set, termed as novel interesting classes, and the human-provided annotations for those novel classes are few, making most existing state-of-the-art object detectors [4]–[8] ineffective.

There have been some prior arts studying few-shot object detection (FSOD) [9]–[14], where detectors are designed to possess strong generalization capabilities, enabling them to detect novel classes with just a few labeled samples. Despite promising results, for the previous algorithms, the number and the member of the object classes that can be detected is fixed after finetuning, while for robots’ exploration tasks, the interesting objects are online defined by a human user, where

¹Bowen Li is with the School of Mechanical Engineering, Tongji University, 201804 Shanghai, China. This work is completed when Bowen Li served as an intern in the Robotic Institute Summer Scholar, Carnegie Mellon University.

²Chen Wang and Sebastian Scherer are with the Robotic Institute, the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA.

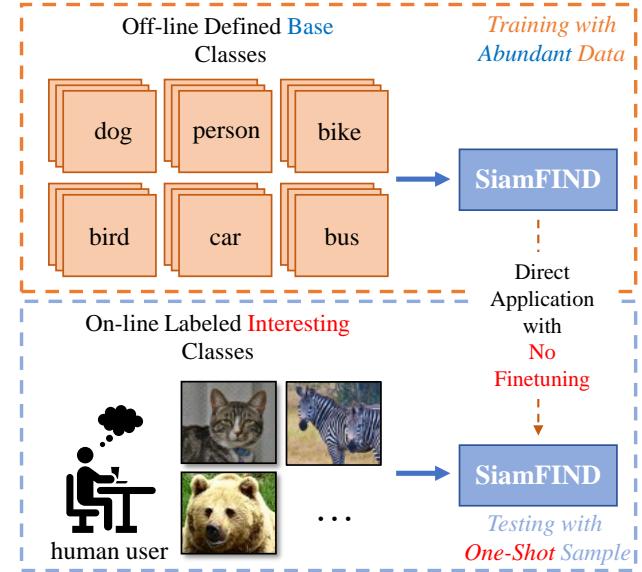


Fig. 1. Illustration of the workflow of proposed SiamFIND. After base training with abundant images, SiamFIND can fast generalize to online deployment without finetuning, where only one-shot samples of the interesting classes are provided.

the novel objects are dynamic and different. In this case, the detectors have to be re-finetuned every time the novel classes are changed, which is neither efficient nor intelligent. Moreover, the provided annotations are even more scarce during real-world implementation (usually only one or two samples of each novel class are provided), broadening the gap between existing arts and expected applications.

To this end, this work mainly focuses on building a dynamic one-shot informed detector (SiamFIND), which is capable of detecting online defined one-shot novel classes without abominable finetuning. Built upon the popular two-stage detector Faster RCNN [4], SiamFIND is also inspired by Siamese visual trackers [15]–[17], which design Siamese network structure to search for a target with the template. Similarly, SiamFIND utilizes an auxiliary dual branch to extract class prototypes from the provided one-shot labeled image (support image). With the class prototypes and proposals from region proposal network (RPN) [4], SiamFIND learns a novel prototype embedding network (PEN) to achieve class-related location feature of the test image (query image), which can be utilized to precisely localize each object. Meanwhile, a learnable cosine similarity (LCS) module is put forward to robustly classify the proposals with prototypes.

Surprisingly, SiamFIND is insensitive to class variations by virtue of the proposed PEN and LCS, compared with other prior works. In other words, after large-scale base

training, the human user only needs to input his interesting objects with only one single sample per class to SiamFIND, detection of these objects is quickly achieved on following captured images, regardless of the numbers or members of these classes. Fig. 1 demonstrates such workflow of SiamFIND.

To validate the effectiveness and superiority of SiamFIND, this work built up a new evaluation setting using COCO dataset [18], where the 80 classes are randomly split into 6 groups (with different members and numbers each group). Provided single labeled image per class, SiamFIND can perform favorable detection in the testset even better than [4], which is slowly finetuned. The experimental results strongly proved the excellent generalization ability and splendid robustness of SiamFIND when faced with class variation issues, which can largely ensure and enable robots' autonomous exploration tasks.

To sum up, the contributions of this work are three-fold:

- This work proposed a novel few-shot detector, termed as SiamFIND, which achieved promising results in detection novel classes without arduous finetuning.
- This work put forward a brand-new Prototype Embedding Network (PEN) and Learnable Cosine Similarity (LCS) module, which learns class agnostic relation to guarantee the outstanding generalization ability of SiamFIND.
- This work constructed a new evaluation benchmark, where novel classes for the COCO dataset are split into different groups to validate the superiority of SiamFIND.

II. RELATED WORKS

A. Object Detection

The task of object detection [4], [7], [8] is to find out all the interesting objects in the image and determine their categories and positions, which is one of the core problems in the field of computer vision.

Object detection algorithms based on deep learning are mainly divided into two categories: two-stage methods [4], [19], [20] and one-stage methods [5]–[8].

For the two-stage target detection algorithm, the interesting region is firstly generated (R-CNN) [4], termed as region proposals (or anchors), then the proposals are classified and regressed for fine results. As the best known two-stage detector, Faster RCNN [4] proposed region proposal network (RPN) based on Fast RCNN [19], which greatly improved the speed of object detection. Based on Faster RCNN, R-FCN [20] proposed to utilize a full convolutional network (FCN) for shared computation and further boosts the speed.

The one-stage target detectors directly extract features from the image to predict object class and location without anchor generation. Among them, the popular SSD [5] algorithm proposes a small convolution filter to predict the class and the offset of the bounding box on the feature map. YOLO [6] takes object detection as a regression problem. Based on a single end-to-end network, the input from the

original image to the output of object position and category is directly completed.

Although the above detection algorithm has achieved considerable detection accuracy, its huge defect is that a huge amount of data are needed for off-line training, where the number of samples in each class is up to tens of thousands. Moreover, the categories that can be detected by the above detection algorithm have been fixed after training, indicating a huge gap from real robots applications.

B. Few-shot Object Detection

After base classes image training, with only a few labeled novel class image samples, few-shot detectors can learn to generalize to the novel classes, without losing the accuracy and robustness of the base classes.

Two main branches are leading in FSOD task, one is based on meta-learning [9], [11]–[13], the other is fine-tuning [10], [14]. For fine-tuning-based few-shot detectors, most parameters of the network are locked after large-scale base training, while only a few parameters are unfrozen. With an appropriate learning rate, the network is fine-tuned with a few samples of both base class and new class. Among them, TFA [10] proposes to fine-tuning only the last two layers of the network (that is, the linear layer of the regression branch and the cosine similarity layer of the detection branch) will achieve considerable results. SRR-FSD [14] proposes that the semantic relationship between novel and base class can assist network to generalize.

There are generally 2 stages of meta learning-based methods [9], [11]–[13]: meta training, meta fine-tune. In the meta training stage, the huge base class is divided into a large number of query images and a small number of support images. The detector learns the meta knowledge from the few support images to detect objects in query images. In the meta tuning stage, the images mixed by a small number of both novel and base classes are also divided into query and support, with a similar training process performed. During the test, support images with annotations are input for the network to manipulate test images, outputting object bounding boxes.

Both 2 kinds of methods can only detect the pre-defined number and members of classes. Faced with dynamic class variation during online deployment, strenuous re-fine-tuning is needed. Besides, existing few-shot detectors all suffer from extreme one-shot cases. Differently, this work proposed SiamFIND, a few-shot interesting object detector that detects novel objects directly after base training with only one-shot information needed.

III. METHODOLOGY

This section first formulate the problem setting, where the task of few-shot object detection and learning schedule of SiamFIND are introduced in detail. Then, the model structure and loss definition of SiamFIND are demonstrated, where the module functions and design motivation are also provided.

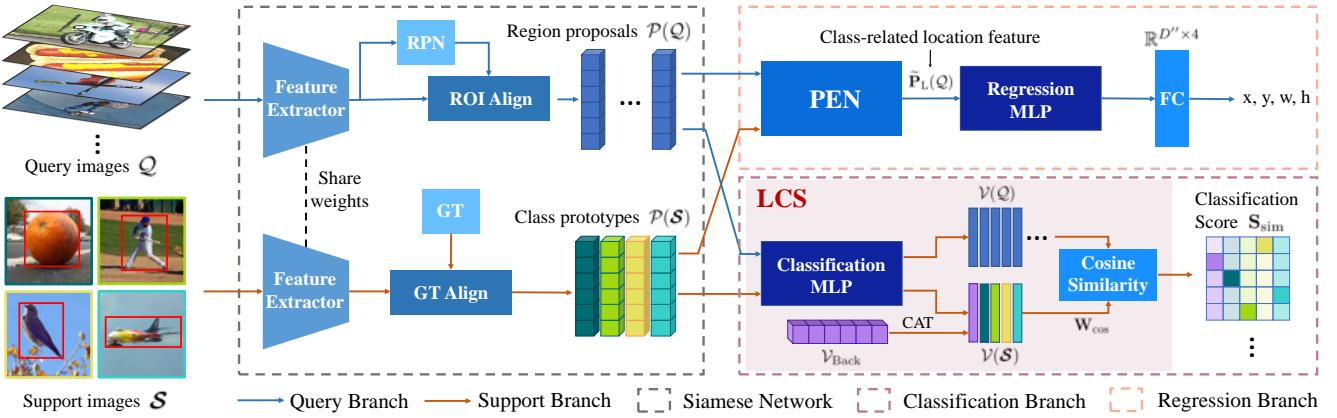


Fig. 2. The overall framework of the proposed SiamFIND. SiamFIND includes 2 branches: query branch and support branch, which are marked out by arrows in different colors. Three modules of SiamFIND, i.e., the Siamese feature extraction network, classification branch, and regression branch are shown by dotted boxes in different colors for clear reference.

A. Problem Definition

This section first formulates the problem setting, where the task of few-shot object detection and learning schedule of SiamFIND are introduced in detail. Then, the model structure and loss definition of SiamFIND is demonstrated, where the module functions and design motivation are also provided.

B. Problem Definition

For few-shot detection tasks, the detectors are firstly trained with abundant images belonging to base class pool \mathcal{B} . Then, given only k shots ($k \leq 10$) images per novel class $n \in \mathcal{N}$, the detector is expected to detect objects belonging to both base classes and novel classes. Since for robots' exploration tasks, the user usually doesn't care about the base classes, rather, they expect the detector to be robust in online defined novel classes $n \in \mathcal{N}$, which is dynamic.

To achieve such tasks, SiamFIND adopts a meta-learning strategy [9], [12], where training and tested are aligned with the episodic paradigm. For the base training stage, in every episodic, a query image Q_b in class b and a support image S_b in the same class are simultaneously input into the model. With proper loss function calculated on the ground truth of Q_b , the network is supposed to learn the meta knowledge from S_b to detect objects in Q_b . The difference between deployment and base training lies in that the ground-truth bounding boxes of class b in Q_b are available only during training.

Different from existing works [9], [12], SiamFIND doesn't need to be finetuned using images in novel classes \mathcal{N} . Once trained, after feeding the supported image $S_n, n \in \mathcal{N}$ into SiamFIND, the model can be directly deployed to detect the novel classes in Q_n regardless of the number and member of \mathcal{N} .

C. Network Structure

To build a few-shot detector without finetuning, we aim to make the detector class agnostic like visual tracking [15], [16]. In other words, the detector is expected to learn the relation between support and query images for detection,

instead of learning or trying to overfit the feature of a certain class [4], [9], [10], [12].

Fig. 2 offers the overall framework of the proposed SiamFIND, which contains two Siamese branches, i.e., the query branch and support branch. With features of query and support images from backbone and region proposal network (RPN) [4], we proposed (a) learnable cosine similarity (LCS) module for classification and (b) prototype embedding network (PEN) for regression the bounding box of proposals.

1) *Siamese Network*: Given that backbone network and RPN are proved to be class agnostic [4], [12], the 2 modules share weights in the Siamese network of SiamFIND. For a given of query images Q , we define the feature from the backbone as $\mathcal{F}(Q) = \psi(Q) \in \mathbb{R}^{1 \times D \times W \times R}$, where D is the channel depth of the feature map. Similarly, with totally C classes of support images $\mathcal{S} = [\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_C]$, the support feature satisfies $\mathcal{F}(\mathcal{S}) \in \mathbb{R}^{C \times D \times W \times R}$.

Then, with p generated proposals in Q from RPN, proposals features $\mathcal{P}(Q) \in \mathbb{R}^{p \times D \times a \times a}$ are achieved using ROIAlign [21], where a denotes the representation size. Using ground truth from support images, we implements similar align process on the support feature $\mathcal{F}(\mathcal{S})$ and obtained the prototypes $\mathcal{P}(\mathcal{S}) \in \mathbb{R}^{C \times D \times a \times a}$.

With proposal feature $\mathcal{P}(Q)$ and prototype $\mathcal{P}(\mathcal{S})$, SiamFIND implements regression and classification using *prototype embedding network* (PEN), and *learnable cosine similarity* (LCS), respectively.

2) *Regression Branch*: Inspired by [11], information from support images can help design class agnostic regression branch, we proposed a locator network to achieve class-related location feature, as presented in Fig. 3.

The proposals $\mathcal{P}(Q)$ are first repeated to $\mathbf{P}(Q) \in \mathbb{R}^{p \times C \times D \times a \times a}$ to match the class number C . Then, we calculate the proto kernels $\mathbf{K}(\mathcal{S}) \in \mathbb{R}^{C \times D \times 1}$ as:

$$\begin{aligned} \mathbf{C} &= \text{conv}(\mathcal{P}(\mathcal{S})) , \\ \mathbf{K}(\mathcal{S}) &= \text{view}(\mathbf{C})\mathbf{W}_1\mathbf{W}_2 , \end{aligned} \quad (1)$$

where Conv denotes 3×3 convolution, $\mathbf{W}_1 \in \mathbb{R}^{(a \times a) \times D'}$, $\mathbf{W}_2 \in \mathbb{R}^{D' \times 1}$ indicates two fully connected

layer, respectively. D' is a hyperparameter, which is set to 1024 in our experiments. With the class proto kernels, class related location feature $\tilde{\mathbf{P}}_L(\mathcal{Q})$ are achieved:

$$\tilde{\mathbf{P}}_L(\mathcal{Q}) = \mathbf{P}(\mathcal{Q}) \odot \mathbf{K}(\mathcal{S}), \quad (2)$$

where \odot denotes convolution.

After the locator network, we follow the design in [4], where regression MLP and a linear are used to get the precise bounding box of proposals $\mathcal{P}(\mathcal{Q})$.

Remark 1: Differently, since $\mathbf{P}_L(\mathcal{Q})$ already possesses class information, the last Linear layer of regression \mathbf{W}_R takes the dimension $\mathbb{R}^{D'' \times 4}$ instead of $\mathbb{R}^{D'' \times (4 \times C)}$, where D'' is the output channel dimension of MLP layer. This allows SiamFIND to fit into any number of classes C to be detected by virtue of the adaptive class dimension of the locator network.

3) *Classification Branch*: According to [10], cosine similarity is an effective method to minimize the intra-class variance, which shows its efficacy in classification. However, we found that merely adopting cosine similarity can barely work, since the background proposals will also achieve a high similarity score, disturbing the filtering process.

To this end, SiamFIND proposed a learnable cosine similarity module. Let $\mathcal{V}(\mathcal{Q}) \in \mathbb{R}^{p \times D''}$ and $\mathcal{V}(\mathcal{S}) \in \mathbb{R}^{C \times D''}$ denote the feature vector of proposals in query image and prototype feature vector of support images, respectively. Classification branch aims to learn a background vector $\mathcal{V}_{\text{Back}} \in \mathbb{R}^{1 \times D''}$, which is concatenated with $\mathcal{V}(\mathcal{S}) \in \mathbb{R}^{C \times D''}$ to get the cosine similarity weights \mathbf{W}_{cos} :

$$\mathbf{W}_{\text{cos}} = \text{CAT}(\mathcal{V}_{\text{Back}}, \mathcal{V}(\mathcal{S})) \in \mathbb{R}^{(C+1) \times D''}, \quad (3)$$

Then, elements $s_{i,j}$ in similarity score matrix \mathbf{S}_{sim} is obtained using:

$$s_{i,j} = \frac{\mathcal{V}_i^\top(\mathcal{Q}) \mathbf{W}_{\text{cos},j}}{\|\mathcal{V}_i(\mathcal{Q})\| \|\mathbf{W}_{\text{cos},j}\|}, \quad (4)$$

where $i \in p$ indicates the i -th proposal, and $j \in (C + 1)$ denotes the j -th weight vector of \mathbf{W}_{cos} .

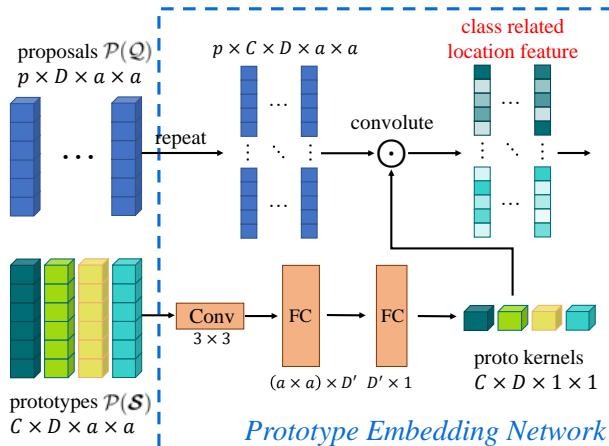


Fig. 3. Details of the Prototype Embedding Network (PEN). With the proposed PEN, class-related location feature is obtained, which can be utilized to achieve bounding box regression for precise localization.

Remark 2: Unlike [4], [10], [12], the LCS in SiamFIND outputs dynamic classification score matrix, *i.e.*, the dimension of \mathbf{S}_{sim} is only decided by the number C of support images \mathcal{S} , while prior work usually off-line defines this dimension, making the detector static.

D. Training Loss

Following [4], [10], [12], the training loss consists of 3 parts, *i.e.*, the RPN loss \mathcal{L}_{RPN} , regression loss \mathcal{L}_{reg} , and classification loss \mathcal{L}_{cls} . For \mathcal{L}_{RPN} and \mathcal{L}_{reg} , the definition has already given in [4], which also proves to work well in SiamFIND. However, for \mathcal{L}_{cls} , using cross entropy loss in [4] won't work well since: (a) elements in \mathbf{S}_{sim} are already probability scores, instead of probability logits; (b) cross entropy loss will end in extremely unbalanced loss of positive and negative proposals, *i.e.*, the positive proposals will be much higher than that of negative proposals. To this end, we proposed a weighted normalize null loss to handle \mathcal{L}_{cls} .

1) *Classification Loss*: Let $\mathcal{V}_{\text{pos}}(\mathcal{Q})$ and $\mathcal{V}_{\text{neg}}(\mathcal{Q})$ denote the feature vector of positive and negative proposals, respective. Their corresponding score vectors are obtained as $\mathbf{S}_{\text{sim},\text{pos}}$ and $\mathbf{S}_{\text{sim},\text{neg}}$ using Eq. (4). Assume the label vector of positive and negative samples are \mathbf{L}_{pos} and \mathbf{L}_{neg} . The classification loss \mathcal{L}_{cls} is given as:

$$\begin{aligned} \mathcal{L}_{\text{cls}} = & \alpha \text{NLL}(\log(\text{norm}(\mathbf{S}_{\text{sim},\text{pos}})), \mathbf{L}_{\text{pos}}) \\ & + \text{NLL}(\log(\text{norm}(\mathbf{S}_{\text{sim},\text{neg}})), \mathbf{L}_{\text{neg}}), \end{aligned} \quad (5)$$

where NLL denotes nll loss and norm indicates normalize operation. α is a hyper parameter to control the influence of positive proposals.

IV. EXPERIMENTS

To validate the efficacy of the proposed SiamFIND in dynamic one-shot object detection scenes, this work mainly

TABLE I
OBJECT AND IMAGES DISTRIBUTION DURING TRAINING AND EVALUATION. DURING EVALUATION, DIFFERENT GROUPS HAVE DIFFERENT MEMBER AND NUMBER OF CLASSES.

Phase	Split	Classes	Images
Training	Pascal VOC split 1	aeroplane, bicycle, boat, bottle, car, cat, chair, diningtable, dog, horse, person, pottedplant, sheep, train, tvmonitor	6135
	COCO class group 1	person, motorcycle, bus, truck, boat, traffic light	2944
	COCO class group 2	stop sign, bench, bird, handbag, suitcase, snowboard, kite, baseball glove, chair	1418
	COCO class group 3	dining table, toilet, laptop, keyboard, cell phone, oven, book	1190
	COCO class group 4	clock, vase, hair drier, refrigerator, bed, sandwich, bird, bench	978
	COCO class group 5	laptop, snowboard, bench, keyboard, bed, sandwich, bird, stop sign, book	1042
Evaluation	COCO class group 6	clock, bus, boat, refrigerator, cell phone, suitcase, bird, motorcycle	1117

TABLE II

OVERALL PERFORMANCE OF THE DETECTORS IN COCO CLASS GROUPS. IN ALL GROUPS, SIAMFIND ACHIEVES BETTER RESULTS THAN OUR BASELINE IN THE ONE-SHOT SCENE. BESIDES, WHILE BASELINE AND FSDET NEED AT LEAST 10 EPOCHS FINE-TUNING, SIAMFIND CAN BE DIRECTLY DEPLOYED ONCE TRAINED. **RED** DENOTES THE BEST RESULTS AMONG 3 DETECTORS IN 1 SHOT CASE.

AP ₅₀ of baseline, FSdet, and SiamFind in COCO new setting									
GroupDetector	Converge Epoch	Shots	Group 1	Group 2	Group 3	Group 4	Group 5	Group 6	
SiamFind	0	1	3.8	0.6	1	0.6	0.8	2.9	
FSdet [12]	10	1	1.1	0.2	0.4	0.1	0.3	0.6	
baseline [4]	30	1	3.5	0	0.5	0	0	1.6	
baseline	30	5	7.1	1	1.8	3.7	3	9.7	
baseline	30	10	9.5	0.9	5.7	5.3	4.3	10.2	

conducts experiments using classes in COCO dataset [18] as test images. To maintain fairness, the baseline method [4], state-of-the-art (SOTA) few-shot detector FSdet [12], and SiamFind are all trained using Pascal VOC split 1 [22].

A. Implementation Details

1) *Details of SiamFind*: SiamFind is built on Faster RCNN [4], adopting MobileNet.v2 [23] as feature extractor network. During training, 4 NVIDIA TITAN X GPU are utilized in parallel, where the model was trained 30 epochs using momentum SGD. For the training process, we adopted a base learning rate of 0.005, with momentum as 0.9 and weight decay of 0.33. p is set to 1024 per image, D is 1280, D' and D'' are both set to 1024. Representation size a of RPN is set to 7. For other implementation details of SiamFind, we followed the original torchvision official code of Faster RCNN.

2) *Evaluation Dataset*: TABLE I represents the details of the dataset utilized in this work. To maintain fair comparison with [12], all models are trained using the same training set, *i.e.*, Pascal VOC split 1, containing a total of 6135 images. For evaluation, since we aim to evaluate the generalization ability of the models, the evaluation is extended in COCO dataset [18]. This work randomly selected 6 groups of classes in total 80 classes of COCO, which is shown in Tab. I. Note that we use one shot per class in MSCOCO training set

TABLE III

PER CLASS RESULTS IN COCO CLASS GROUP 1. THE NOVEL CLASSES AND OUTSTANDING RESULT OF SIAMFIND ARE MARKED OUT IN **RED**.

Group	Class Group 1					
Class	person	motorcycle	bus	truck	boat	traffic light
baseline	3.45	0.00	17.28	0.00	0.00	0.00

SiamFind

12.30	3.56	4.33	1.45	1.29	0.00
--------------	-------------	-------------	-------------	-------------	------

TABLE IV

PER CLASS RESULTS IN COCO CLASS GROUP 6. THE NOVEL CLASSES ARE MARKED OUT IN **RED**.

Group	Class Group 6							
Class	clock	bus	boat	refrigerator	cell phone	suitcase	bird	motorcycle
baseline	0.0	1.0	0.0	0.8	0.1	3.0	1.0	0.0

0.06	10.3	3.4	0.2	0.07	0.8	2.4	6.3
-------------	-------------	------------	------------	-------------	------------	------------	------------

randomly as a support set and all images in the MSCOCO eval set as evaluation.

B. Overall Performance and Analysis

The overall performance in COCO novel class groups is presented in TABLE II. We consider our baseline as Faster RCNN [4], which is finetuned in each group with corresponding shots following [10]. Proposed SiamFind can achieve better results than baseline and a SOTA few-shot detector FSdet [12]. Moreover, while baseline method and FSdet [12] needs at least 30 epochs fine-tuning, SiamFind can be directly applied on the novel classes with the only one-shot sample.

TABLE III and TABLE IV display per class results of SiamFind and baseline of group 1 and group 6, respectively. Surprisingly, when baseline can't work well in novel classes like a truck, bus (group 6), bird, motorcycle, *etc.*, with a single shot per class, SiamFind can get better result even without fine-tuning.

V. FUTURE WORK

From both Tab. II and Tab. III, despite SiamFind is better than simply finetuning method [4], its mAP is still very low, which guides us to the inspections. According to our experiments, the loss of RPN \mathcal{L}_{RPN} and regression branch \mathcal{L}_{reg} maintain robust when novel classes are introduced, which illustrates that the RPN and regression branch can achieve favorable class agnostic. Nonetheless, the main hardship lies in the classification branch, the loss of which vibrates violently with novel classes, indicating that the classifier still can't live up to our expectation, which is to learn class agnostic meta knowledge during base training. The above observations lead us to the following future work:

- **Inspect feature extraction:** For the failure of classification, one assumption is that the feature of novel classes is not robust. In other words, the feature from the same novel class may vary significantly, making the inferior robust classification.
- **Design more robust classifier:** The classifier is currently based on cosine similarity, which can be improved to some learnable deep strategy like relation network in [24], [25].

- Classification loss definition: Another reason for the unsatisfying performance of the classification is that the positive proposals can achieve a high background score, indicating that the current loss definition still not focuses on positive samples enough, which can be further improved.

VI. CONCLUSION

This work presents a novel few-shot object detector, which, once trained, maintains its robustness when one-shot novel classes enroll without further updating. Specifically, a prototype encoding network is proposed for class agnostic regression and a learnable cosine module is put forward for robust classification. Experiments conducted on 6 groups of COCO novel classes demonstrate the superiority. Despite better results, the performance of SiamFIND is still far from enough to be deployed on real robots, whose defects and future work are also summarized.

ACKNOWLEDGMENT

I extend my sincere gratitude for *Prof.* Sebastian Scherer and *Dr.* Chen Wang, who provided me invaluable suggestions and guidance in this work. I'm also thankful for all the members of the Airlab, who offered their indispensable help to me. This work is supported by the Robotic Institute, Carnegie Mellon University.

REFERENCES

- [1] H. Karaoguz and P. Jensfelt, "Object Detection Approach for Robot Grasp Detection," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 4953–4959.
- [2] R. Reid, A. Cann, C. Meiklejohn, L. Poli, A. Boeing, and T. Brauni, "Cooperative Multi-robot Navigation, Exploration, Mapping and Object Detection with ROS," in *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 1083–1088.
- [3] S. Gidaris and N. Komodakis, "Locnet: Improving Localization Accuracy for Object Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 789–798.
- [4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks," *Advances in Neural Information Processing Systems*, vol. 28, pp. 91–99, 2015.
- [5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot Multibox Detector," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 21–37.
- [6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-time Object Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.
- [7] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.
- [8] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.
- [9] X. Yan, Z. Chen, A. Xu, X. Wang, X. Liang, and L. Lin, "Meta R-CNN: Towards General Solver for Instance-Level Low-Shot Learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9577–9586.
- [10] X. Wang, T. E. Huang, T. Darrell, J. E. Gonzalez, and F. Yu, "Frustratingly Simple Few-Shot Object Detection," *arXiv preprint arXiv:2003.06957*, 2020.
- [11] J.-M. Perez-Rua, X. Zhu, T. M. Hospedales, and T. Xiang, "Incremental Few-Shot Object Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 13 846–13 855.
- [12] Y. Xiao and R. Marlet, "Few-Shot Object Detection and Viewpoint Estimation for Objects in the Wild," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 192–210.
- [13] J. Wu, S. Liu, D. Huang, and Y. Wang, "Multi-Scale Positive Sample Refinement for Few-Shot Object Detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020, pp. 456–472.
- [14] C. Zhu, F. Chen, U. Ahmed, Z. Shen, and M. Savvides, "Semantic Relation Reasoning for Shot-Stable Few-Shot Object Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 8782–8791.
- [15] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-Convolutional Siamese Networks for Object Tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016, pp. 850–865.
- [16] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High Performance Visual Tracking with Siamese Region Proposal Network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8971–8980.
- [17] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4282–4291.
- [18] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014, pp. 740–755.
- [19] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1440–1448.
- [20] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object Detection via Region-Based Fully Convolutional Networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 379–387.
- [21] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2017.
- [22] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.
- [23] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted Residuals and Linear Bottlenecks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.
- [24] L. Zhang, S. Zhou, J. Guan, and J. Zhang, "Accurate Few-Shot Object Detection With Support-Query Mutual Guidance and Hybrid Loss," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 14 424–14 432.
- [25] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, "Learning to Compare: Relation Network for Few-Shot Learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 1199–1208.

Self-learning of Crawling Behavior for a Modular Quadruped Robot with Bayesian Optimization

Chuan Li, Lu Li

Abstract—Modular robots are intuitively designed for quick adaptation under various environments. The control method alters along with the configuration change of module combination. Implementing such algorithms requires complicated spatial geometry calculation and different moving properties of modules. Based on common locomotion patterns of the quadruped, we use Bayesian Optimization to conduct a parametric search towards a feasible controller. We perform several experiments to measure the optimizing performance during crawling learning, and our results show that the Bayesian Optimization algorithm is capable of creating an efficient learning process toward a locomotive gait. A high exploration factor and a large vertical amplitude are both beneficial to the optimal parameters searching.

Index Terms—Quadruped locomotion, Gait optimization, Parametric search

I. INTRODUCTION

Compared to traditional wheeled robots that can only travel on plain grounds, modular robots have stronger mobility because of the ability to overcome rough terrains with vertical movement of the legs. However, legged locomotion is much more difficult to model and control for its complexity. Although several controllers for quadruped robots have been proposed [1] [2] [3], the configuration and tuning of their parameters remain a challenging problem. It is extremely difficult to tune these parameters manually, especially when the robot is operating in the physical world with unpredictable noise. Therefore, parametric search for the gait controllers has become the key to control the locomotion of legged robots.

Various machine learning and optimization methods have been utilized to find these parameters. Such approaches include genetic algorithms [4], Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [5], gradient descent methods [6], and Bayesian Optimization (BO) [7] [8] [9], etc. BO is a sample-efficient black-box global optimization algorithm that is especially suitable when the objective function is expensive to evaluate [10], which is usually the case in the context of robotics. Performing experiments with robots are usually time-consuming and wearing. BO hence has more advantages over the other methods, especially for modular robots.

Most of the previous studies on gait optimization using BO focus on the influence of different configurations of the BO algorithm itself. For example, [8] incorporates domain knowledge to reduce dimensionality for BO in higher dimensions of the parameters. [9] proposes an Alternating Bayesian Optimization (ABO) algorithm that iteratively learns the parameters through interactive trials, resulting in sample

efficiency and fast convergence. In this work, we focus on the influence of the BO exploration factor and the vertical amplitude of robot crawling.

In this paper, we assess different exploration factors and locomotion settings to our robot and use Bayesian Optimization to conduct a parametric search towards a locomotive controller. In our experiments, the robot tries to find suitable controllers parameters to move ahead.

The remainder of the paper is organized as follows. Section II overviews the system and describes the proposed methodology in detail. Section III evaluates the methodology by experiment results, and discusses on potential future work. Lastly, section IV concludes the paper.

II. METHODS

In this work we assemble a quadruped robot (shown in Fig.1) to test a self-learning process of a locomotive controller.

A. Robot Assembly

The modular robot project is called EigenBot, which currently enables simulated modules including bendy module (the module attached directly to the body), elbow module, foot module, etc. All four legs are centrosymmetric to the body. Each of them only contains the first three modules and has 3 degrees of freedom (DOF). The elbow module and the foot module are static, while the bendy module can bend within 180°.



Fig. 1. Quadruped robot: each leg has 3 bendy modules, which enables the foot to move in 3 DOF.

B. Quadruped Crawling Controller

The controller of the robot orders each joint to move in a sine wave described in Equation.1.

$$\theta_{L|R,i} = A_i \sin \left(\frac{t}{T} + \phi_{L|R,i} \right) \quad (1)$$

θ denotes the target angle of a joint (in degree). A denotes the amplitude of the wave (in degree). Specifically, A_2 affects the overall vertical amplitude of the robot movement. t is the simulation time in second and period T is unified to 0.3s (this period will be longer in the simulation due to communication delay). ϕ denotes the initial phase in radian. Note that the subscript 1 to 3 denote hip joint to ankle joint respectively.

To reduce the parameter space of BO, we also synchronize the diagonal legs and similarize all joints' wave amplitude. The final parameter space is shown in Table.I.

TABLE I
PARAMETER SETTINGS OF THE CRAWLING CONTROLLER

Parameters	Range
A_1	$[0, 25]$
A_2	$h \cdot [0, 25]$
A_3	$[0, 25]$
$\phi_{L,1}, \phi_{L,2}, \phi_{L,3}$	$[-\pi, \pi]$
$\phi_{R,1}, \phi_{R,2}, \phi_{R,3}$	$[-\pi, \pi]$

In our experiment, vertical amplitude factor $h \in \{1.0, 1.5\}$. These 9 parameters will be the input of the BO algorithm.

C. Bayesian Optimization

Bayesian Optimization uses Gaussian Process to create a probabilistic surrogate model of the unknown objective function. Gaussian Process is a collection of random variables, any finite number of which have (consistent) joint Gaussian distributions [11]. It can be defined by its mean function $m(x)$ and covariance function $k(x, x')$, and represented as Equation.2.

$$f \sim \mathcal{GP}(m, k) \quad (2)$$

In each iteration, BO samples a point selected by the acquisition function and updates the surrogate model created by GP until a certain number of iterations is reached. We used Upper Confidence Bound (UCB) as the acquisition function of BO and implemented the algorithm with Python. The exploration factor is tested by two values, that is, $\kappa \in \{2.5, 30.0\}$. The objective function in our experiments is set to be the forward distance passed by the robot.

D. Experimental Setup

We use CoppeliaSim as the robot simulator (previously called V-REP), which is based on a distributed control architecture: each object/model can be individually controlled via an embedded script, a plugin, a ROS or BlueZero (BO) node, a remote API client, or a custom solution. We choose the BO-based remote API to apply position control on joints via Python scripts. We run 50 iterations to obtain a result of parameters combination from the BO algorithm, and each iteration lasts 50 seconds for our robot to crawl. Each joint is set with a maximum torque of 100 N·m.

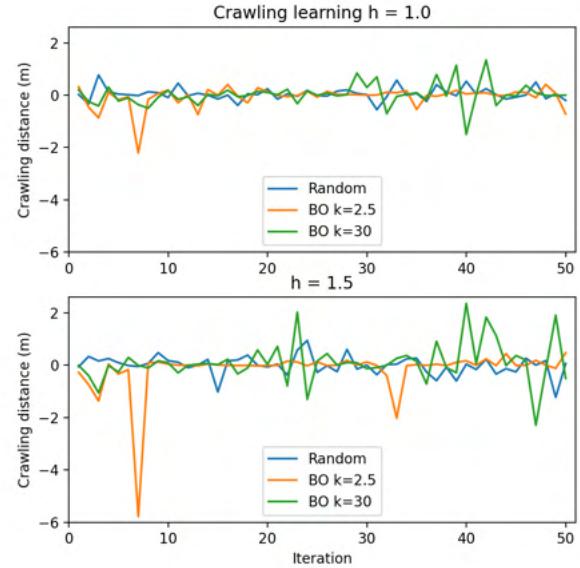


Fig. 2. Learning process: h denotes the vertical amplitude factor. κ denotes the exploration factor of acquisition function in BO algorithm.

TABLE II
LONGEST CRAWLING DISTANCE (M)

	Random	BO $\kappa = 2.5$	BO $\kappa = 30$
$h = 1.0$	0.77	0.41	1.34
$h = 1.5$	0.95	0.48	2.36

III. RESULTS AND DISCUSSION

As the vertical amplitude factor h increases, the landing feet step harder, providing more support to move forward, while the lifting feet release stress from the ground, resulting in less friction resistance. These lead to a more oscillatory learning process shown in Fig.2, which is beneficial for the exploration to optima. Moreover, a low exploration factor κ is harmful to the learning, ending up with results even worse than those of random exploration displayed in Table.II.

The line patterns in Fig.2 also enhance the consistency of the BO learning process. At around the 7th iteration, both BO with $\kappa = 2.5$ and BO with $\kappa = 30$ explore to an extremely low value. Their optimal learning results both appear around the 40th iteration, along with several sub-optimal values.

Further BO experiments can be established on EigenBot with similar configurations. It is vital to restrain the parameter space by synchronizing certain joints or simplifying its structure, otherwise, it would be expensive to find optima. On the other hand, it is also interesting to assemble an EigenBot with few joints but it has a novel structure/configuration, something manual control could hardly be feasible yet self-learning algorithms would easily manage. Moreover, the same robot learning to move on different terrain is another aspect of studies.

IV. CONCLUSIONS

In this work, we assembled a modular quadruped robot, implemented a sine wave controller with a few variable parameters to be decided by the Bayesian Optimization algorithm. We found that both a high exploration factor from the algorithm side or a large vertical amplitude factor from the physics side can greatly aid the crawling learning process. Additionally, this learning algorithm can be applied to various configurations of modular robots. While simulations are a powerful tool to explore controllers before deploying those to real robots, roboticists should be aware that those will not work in the same manner that they performed inside simulations.

ACKNOWLEDGMENT

Chuan sincerely thanks Howie Choset, Lu Li, and Jaya Aadityaa for their continuous assistance in the Biorobotics lab. Appreciate Rachel and John for concern and coordination between scholar and lab. Thank Andre for the previous lab internship which started the research experience.

REFERENCES

- [1] A. J. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," *Neural Networks*, vol. 21, no. 4, pp. 642–653, 2008, robotics and Neuroscience. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0893608008000804>
- [2] S. Coros, A. Karpathy, B. Jones, L. Reveret, and M. van de Panne, "Locomotion skills for simulated quadrupeds," *ACM Transactions on Graphics*, vol. 30, no. 4, p. Article TBD, 2011.
- [3] M. H. Raibert, *Legged Robots That Balance*. USA: Massachusetts Institute of Technology, 1986.
- [4] S. Chernova and M. Veloso, "An evolutionary approach to gait learning for four-legged robots," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, 2004, pp. 2562–2567 vol.3.
- [5] N. Hansen, "The cma evolution strategy: a comparing review," *Towards a new evolutionary computation*, pp. 75–102, 2006.
- [6] R. Tedrake, T. W. Zhang, H. S. Seung, *et al.*, "Learning to walk in 20 minutes," in *Proceedings of the Fourteenth Yale Workshop on Adaptive and Learning Systems*, vol. 95585. Beijing, 2005, pp. 1939–1412.
- [7] D. J. Lizotte, T. Wang, M. H. Bowling, D. Schuurmans, *et al.*, "Automatic gait optimization with gaussian process regression." in *IJCAI*, vol. 7, 2007, pp. 944–949.
- [8] A. Rai, R. Antonova, S. Song, W. Martin, H. Geyer, and C. G. Atkeson, "Bayesian optimization using domain knowledge on the atrias biped," 2017.
- [9] K. Yuan, I. Chatzinikolaidis, and Z. Li, "Bayesian optimization for whole-body control of high-degree-of-freedom robots through reduction of dimensionality," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2268–2275, 2019.
- [10] E. Brochu, V. M. Cora, and N. de Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," 2010.
- [11] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer school on machine learning*. Springer, 2003, pp. 63–71.

Understanding and Predicting Human Activities in Search and Rescue Tasks

Ruiyu Li¹, Yue Guo², Dana Hughes², Katia Sycara²

Abstract—In an urban search and rescue (USAR) task, the existing preconditions for successful triage and the complexity of environments may impair rescuers’ performance. To build an artificial agent that can assist rescuers in performing their tasks efficiently, it is crucial to understand the correct sequence of activities required to complete a task and predict rescuers’ high level strategy. While previous efforts have allowed agents to predict rescuers’ navigation and victim triage strategy, there is no work tackling rescuers’ activities together at a high abstraction level. This work proposes steps towards this by capturing high level behaviors in USAR tasks from training hierarchical reinforcement learning agents. (1) We learn an expert strategy module by training an optimal agent with portable options. (2) We utilize LSTM networks to predict rescuers’ high level strategy at each time-step. Early-stage experiment results show the capability of the AI agent developed by our method to optimally perform USAR tasks and accurately predict rescuers’ activities. This could be useful for enabling agents to provide intervention and further extend to multi-agent scenarios.

Index Terms—Search and Rescue Robots, Reinforcement Learning

I. INTRODUCTION

Consider a USAR scenario where rescuers try to save as many victims as possible in a building damaged during an earthquake in limited time. Victims with varying degrees of injury would require different numbers of rescuers to save and structural perturbations in the building could bring difficulty to rescue activities. For instance, a medic navigates to a victim that is buried beneath rubble, but before the medic can save the victim, the medic needs to wait for an engineer to remove the rubble first. While human rescuers may fail to recognize the appropriate activity sequence required to complete a task and thus cause inefficiency, an artificial agent that observes, predicts, and intervenes into their navigation and rescue activities can correct their inefficient behaviors and help rescuers improve performance.

Previous efforts such as [1] [2] have developed agents with the ability to predict rescuers’ navigation and victim triage strategy separately. Differently, we investigate rescuers’ various high-level activities together to capture their temporal relationships. This paper provides initial results of learning expert strategy of high-level activities which can serve as the knowledge of the artificial agent and predicting the activities of artificial rescuers.

¹Ruiyu Li is with the Department of Computer Science and Engineering, University of Michigan, Ann Arbor, MI 48109 USA
ruiyuli@umich.edu

²Yue Guo, Dana Hughes and Katia Sycara are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA {yueguo, danahugh, sycara}@andrew.cmu.edu

Reinforcement learning (RL) provides methods to train agents interacting with the environment to solve various kinds of tasks, such as video games [3], autonomous driving [4], and business management [5]. Executing state-action-next state transitions and receiving rewards, the RL agent learns a policy by exploring the state and action spaces corresponding to the task. While in long-horizon tasks standard RL generally results in agents with poor performance, hierarchical reinforcement learning (HRL) decomposes a long-horizon RL task into a hierarchy of sub-tasks consisting of a sequence of low-level actions to effectively reduce the task’s long horizon [6]. The HRL agent learns a higher-level policy that optimally chooses sub-tasks as high-level actions to execute. Since our focus is on rescuers’ high-level activities like approaching a victim and navigating to a room instead of primitive actions like moving forward and turning left, it is natural to model rescuers as HRL agents.

Specially, we utilize the option framework [7] to simulate artificial rescuers, where each option represents an activity of a rescuer that we want to understand and predict. To increase the transferability of the learned policy over options, we follow [8] to train options in both the problem space and the agent space. The option-aware trajectories generated by the artificial rescuers have many potential uses: we can compare them with human’s trajectories and then evaluate how humans perform in USAR tasks; we can resolve the problem of data deficiency and also increase the diversity of human data by simulating various types of agents; we will be able to explain how humans perform by inferring their options.

For any rescuer, our assistant agent observes the states the rescuer encounters and the options it takes. Using these observations as training data, the artificial agent can train a classifier to predict the rescuer’s options at each step. As human rescuers learn from advice given by our assistant agent, the prediction task is challenging because rescuers are constantly learning, which would produce inconsistent data. But still, there are some underlying relationship among rescuers’ behaviors as time passes. Recurrent neural networks (RNN) have been successfully applied to time-dependent prediction [9] and recommendation [10]. We take advantage of Long Short Term Memory (LSTM) networks, a special kind of RNN capable of learning both long-term and short-term dependencies [11], to predict rescuers’ activity at each time step. One challenge of using the LSTM model is that a large amount of data is required to train LSTM networks to obtain good results. Thus, we collect data of option-aware trajectories from simulating varieties of artificial rescuers.

II. PRELIMINARIES

A. HRL Based on Semi-Markov Decision Process (SMDP)

A **SMDP** is a stochastic control process consisting of a state space S of the environment, a set of actions A that can be taken by an agent, a set of sub-tasks O , a transition function P , and a reward function r . An **option** o is defined as a tuple (I_o, π_o, β_o) . $I_o : S \rightarrow \{0, 1\}$ indicates whether an option o can be initialized in each state. $\pi_o : S \times A \rightarrow [0, 1]$ represents the policy of an option o . $\beta_o : S \rightarrow [0, 1]$ gives the probability of an option o terminating in each state.

A two-level HRL based on options can be formed as follows: at any state s , an agent chooses an option according to a high-level policy, takes an action based on the policy of that chosen option, transit to the next state drawn from P , and receives a reward. In general, the problem definition of the HRL is to find an optimal hierarchy policy. In this work, we focus on Markov policies over options, $\mu : S \times O \rightarrow [0, 1]$. Let $\mathcal{E}(o, s, t)$ represent the event of o being taken in state s at time t . We define the value of taking option o in state s under policy μ as

$$Q^\mu(s, o) = E \{ r_{t+1} + \gamma r_{t+2} + \dots | \mathcal{E}(o\mu, s, t)\}$$

, where γ is the discount factor and $o\mu$ is the composition of o and μ denoting the policy that first follows o until it terminates and then initiates μ in the resultant state. Learning Methods such as SMDP Q-learning [12] and intra-option Value Learning [13] are used to obtain the optimal value functions.

B. RNN and LSTM

RNN is a class of Neural Network in which internal units may form a directed loop to demonstrate the state history of previous inputs. The structure of RNN allows the model to store temporal contextual information directly without explicitly defining the length of temporal contexts. Among several versions of RNNs, **LSTM** networks are capable of memorizing sequences with long range temporal dependencies. For a sequence input (x_1, x_2, \dots, x_T) , the hidden state h_t , $t \in \{1, 2, \dots, T\}$, of the RNN is updated by

$$h_t = f(h_{t-1}, x_t)$$

, where f is a activation function. As illustrated in Figure 1, LSTM updates its hidden state sequentially similar to RNN, but LSTM contains the update gate i_t that determines how to update remembered information, the forget gate c_t that decides the amount of remembered information to forget, and the output gate o_t decides the amount of the remembered information to output. The following standard equations describe recurrent algebraic relationship of the LSTM unit:

$$\begin{aligned} f_t &= \sigma(W_f(x_t, h_{t-1})) \\ i_t &= \sigma(W_i(x_t, h_{t-1})) \\ c'_t &= \tanh(W_c(x_t, h_{t-1})) \\ c_t &= i_t \odot c'_t + f_t \odot c_{t-1} \\ o_t &= \sigma(W_o(x_t, h_{t-1})) \\ h_t &= o_t \odot \tanh(c_t). \end{aligned}$$

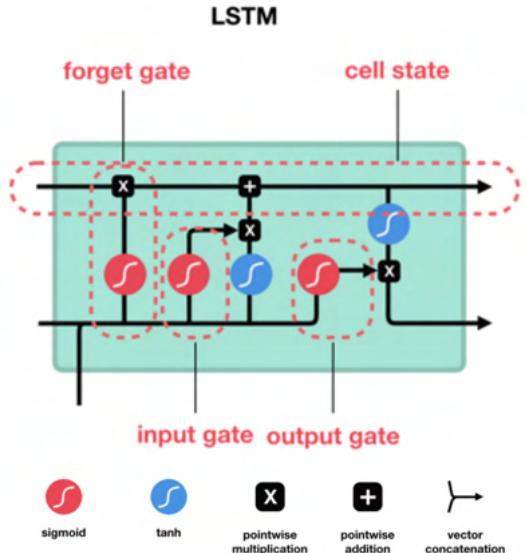


Fig. 1. The repeating module in LSTM [14].

III. APPROACH

A. Environment and task design

We designed a four-room environment for our study, which contains four 6×6 2-D grid rooms, walls, balls, switches and doors with locks, as shown in Figure 2. In each episode, the agent is initialized at a random location in the top-left room and four balls are placed at random locations in the environment to represent victims. There are doors between the two top rooms and the two bottom rooms, which are locked initially and at random locations. Each locked door corresponds to exactly one switch which can be toggled to open the door. Switches are also randomly placed in the area that the agent are able to reach in each episode. Once a door is opened, toggling the switch will have no effect.

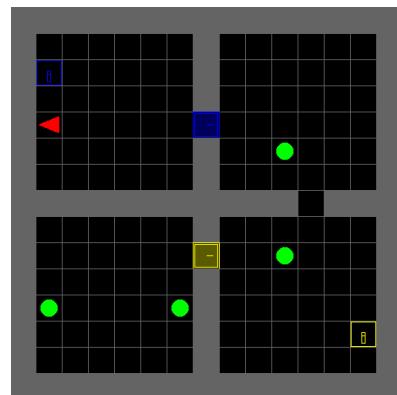


Fig. 2. Four-room environment. In this example, only the blue switch can open the blue door and the yellow switch can open the yellow door

An agent can choose from four actions, FORWARD, TURN-LEFT, TURN-RIGHT, and TOGGLE, to take at each step. In each episode, the goal of the agent is to rescue all

TABLE I
AGENT'S OPTIONS IN THE FOUR-ROOM ENVIRONMENT

Option	initialization Condition	Termination Condition
navigate to a room	agent is not in the room	agent is in the room
go to a switch and toggle it	agent is in the same room as the switch	the switch is toggled
triage a victim	agent is in the same room as the victim	the victim is triaged

the four victims using as few options as possible. The agent cannot go through walls and cannot step upon other objects. Thus, an agent need to navigate to the switch and toggle it before it can navigate to another room with locked door, which is the sequence of activities we expect the agent to learn. Besides, the agent is expected to save all the victims in a room once it enters the room and not waste time to visit already visited rooms.

B. Learn expert knowledge

To extract expert strategy of activities as knowledge to provide intervention, we design options which correspond to activities in USAR tasks and train an optimal fully observable HRL rescuer with the options to learn the correct sequences of activity. The training algorithm is deep Q-network with experience replay and fixed Q-targets [15]. The three fully-connected layers in the deep Q-network is initialized with equal weights and biases. We use both problem space options and agent space options so that the learned knowledge over options is more general and can be applied to more environments [8].

C. Option prediction

For any rescuer, the artificial agent observes the states the rescuer encounters and the options it takes. Using these observations as training data, the artificial agent can train a classifier to predict the rescuer's options at each step. Specially, at each time-step, we predict the option that the agent is going to take in the following step using LSTM. We train the LSTM using data of the last 100 episodes and evaluate the prediction accuracy of LSTM on the next episode. The input to the model is a time sequence of state features $[X_t, X_{t+1}, \dots, X_{t+T}]$ and the label is the option the agent is going to take at time-step $(t+T)$. We also use SVM and MLP whose input is state features X_t at each time-step and label is the option the agent is going to take at each time-step t to serve as baselines.

To collect data for option prediction, we simulate various artificial rescuers to perform USAR tasks by setting $\beta = 0.9, 0.8, 0.75$ and 0.7 separately in the Softmax option selection $\Pr(o|s) = \frac{e^{\beta * Q(s, o)}}{\sum_{o'} e^{\beta * Q(s, o')}}$ to train artificial rescuers. In this way, we obtained four kinds of agents with varied performance in the four-room USAR tasks.

IV. EXPERIMENTS AND RESULTS

A. Learn expert knowledge

To learn expert knowledge of performing USAR tasks involving preconditions, we train a fully observable HRL

agent with predefined rule-based options. We consider three kind of options for an agent, described in Table I. Note that the options are not specifically defined for any fixed environment, but rather it can be applied to any environment that involves those activities. Denote the number of steps the rescuer takes to fulfill the option of triage a victim as n . The agent will receive a reward of $(+2 \times \gamma^n)$ when it saves a victim and $+1$ when it toggles a switch for the first time. The agent will be given a penalty of -1 if its chosen option can't be achieved finally.

We trained the agent for 7000 episodes and evaluated on 1000 episodes. The training curve over three runs is displayed in Figure 3, where the curve converges normally. The average number of options the agent takes in each episode during the evaluation are 10. Since the smallest number of options the agent needs to take to complete a USAR task in the four-room environment is 9 and the locations of each objects vary from episode to episode, it is fair to consider our trained agent as an expert in solving USAR task in the environment.

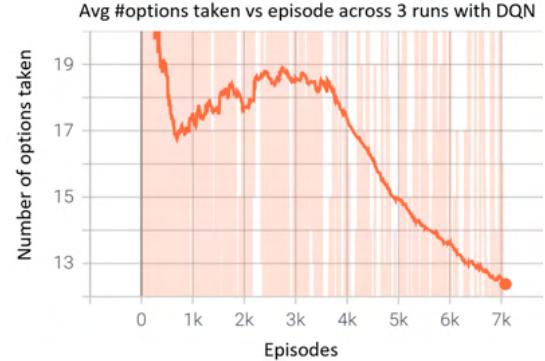


Fig. 3. Average number of options taken - learning curve of the optimal rescuer

B. Simulate artificial rescuers and predict their options

Figure 4 shows the learning curves of two simulated artificial rescuers. From the figures, we can see that they have different performances in solving USAR tasks. The curves converge very slow and the performance of this artificial rescuer are far from optimal, indicating the meaningfulness of providing intervention that we plan to do in the next step.

We conducted the prediction experiment as described in the Approach section using data of the simulated artificial rescuers from episode 2000 to episode 5000 and report the average prediction accuracy across four rescuers in Table II.

The LSTM model outperforms both SVM and MLP in the prediction accuracy.

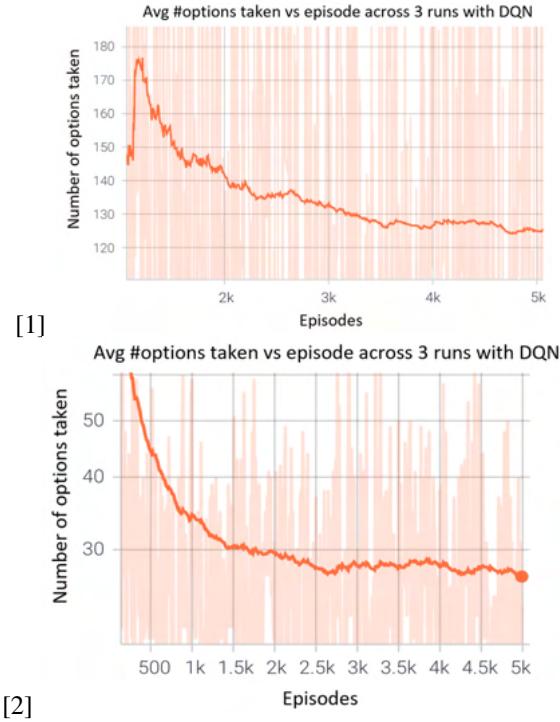


Fig. 4. Average number of options taken - sample learning curves of artificial rescuers

TABLE II
PREDICTION ACCURACY ACROSS FOUR RESCUERS

SVM	MLP	LSTM
0.32	0.58	0.74

V. DISCUSSIONS AND FUTURE WORK

While the LSTM model achieves higher prediction accuracy than both SVM and MLP, models with better performance on the prediction task should be developed because correct prediction of rescuers' options is crucial for giving intervention and we don't want to waste the expert's advice or mislead rescuers.

In the future, we would like to utilize the learned expert knowledge and the prediction ability to provide intervention. Basically, the intervention will be based on the correct identification and prediction of options, and this is helpful for human-machine trust and improve the mission performance in general. To our knowledge, among all the work on transferring knowledge from a teacher to students, teacher-initiated action advising approach is the most relevant to our scenario. Learning Hierarchical Teaching Policies for Cooperative Agents [16] is the only one that advise temporally extended sequences of primitive actions (i.e., high-level strategy) via hierarchical reinforcement learning. But their situation is two RL agents teach each other while learning

to cooperatively complete tasks. In the predictive-advising framework introduced in [17], a RL teacher agent gives advice on actions to RL agents a limited number of times, which could be extended and improved for our purpose to provide advice on options to rescuers. Moreover, extending our approach to multi-agent settings and more complicated environments are interesting directions.

VI. CONCLUSIONS

This work proposes methods to develop an AI agent assisting rescuers in performing USAR tasks more efficiently by recognizing the correct sequences of behaviors required to complete a task and predicting rescuers' high level strategy at each time-step. Initial experiment results show that (1) we successfully learn an expert strategy module by training an optimal HRL agent with portable options and (2) our approach by using LSTM networks to predict rescuers' high level strategy at each time-step outperforms the baseline models, SVM and MLP. This could be further used to enable agents to provide advice on options to rescuers.

ACKNOWLEDGMENT

The authors would like to thank Rachel Burcin, John Dolan, Jennie Piotrkowski, and all of Carnegie Mellon University's Robotics Institute Summer Scholars (RISS) program. Ruiyu Li also wants to thank her mentors, Prof. Katia Sycara, Dr. Dana Hughes and Yue Guo.

REFERENCES

- [1] G. Y., J. R., H. D., L. M., and S. K., "Transfer learning for human navigation and triage strategies prediction in a simulated urban search and rescue task," *IEEE International Conference on Robot and Human Interactive Communication*, 2021.
- [2] V. Jain, R. Jena, H. Li, T. Gupta, D. Hughes, M. Lewis, and K. Sycara, "Predicting human strategies in simulated search and rescue task," *Workshop on Artificial Intelligence for Humanitarian Assistance and Disaster Response, NeurIPS*, 2020.
- [3] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, p. 253–279, Jun 2013.
- [4] J. Chen, B. Yuan, and M. Tomizuka, "Model-free deep reinforcement learning for urban autonomous driving," in *2019 IEEE intelligent transportation systems conference (ITSC)*. IEEE, 2019, pp. 2765–2771.
- [5] Q. Cai, A. Filos-Ratsikas, P. Tang, and Y. Zhang, "Reinforcement mechanism design for fraudulent behaviour in e-commerce," 2018.
- [6] S. Pateria, B. Subagja, A.-h. Tan, and C. Quek, "Hierarchical reinforcement learning: A comprehensive survey," *ACM Comput. Surv.*, vol. 54, no. 5, June 2021.
- [7] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial Intelligence*, vol. 112, no. 1, pp. 181–211, 1999.
- [8] G. Konidaris and A. Barto, "Building portable options: Skill transfer in reinforcement learning," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, ser. IJCAI'07, 2007, p. 895–900.
- [9] J. Martinez, M. J. Black, and J. Romero, "On human motion prediction using recurrent neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [10] R. Devooght and H. Bersini, "Long and short-term recommendations with recurrent neural networks." Association for Computing Machinery, 2017.
- [11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

- [12] R. Parr and S. Russell, “Reinforcement learning with hierarchies of machines,” *Advances in neural information processing systems*, pp. 1043–1049, 1998.
- [13] R. S. Sutton, D. Precup, and S. P. Singh, “Intra-option learning about temporally abstract actions,” in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML ’98, 1998, p. 556–564.
- [14] M. Phi, “Illustrated guide to lstm’s and gru’s: A step by step explanation.” [Online]. Available: <https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>
- [15] M. Roderick, J. MacGlashan, and S. Tellez, “Implementing the deep q-network,” *arXiv preprint arXiv:1711.07478*, 2017.
- [16] D.-K. Kim, M. Liu, S. Omidshafiei, S. Lopez-Cot, M. Riemer, G. Habibi, G. Tesauro, S. Mourad, M. Campbell, and J. P. How, “Learning hierarchical teaching policies for cooperative agents,” *arXiv preprint arXiv:1903.03216*, 2019.
- [17] M. E. Taylor, N. Carboni, A. Fachantidis, I. Vlahavas, and L. Torrey, “Reinforcement learning agents providing advice in complex video games,” *Connect. Sci.*, vol. 26, no. 1, p. 45–63, Jan. 2014.

Generating Stenosis Regions of Interest in X-Ray Coronary Angiography using Deep Neural Networks

Jennifer Liu¹, James K. Miller², Keith Dufendach³ and Artur Dubrawski²

Abstract—Invasive Coronary Angiography remains the gold standard diagnostic tool for Coronary Artery Disease (CAD), the leading cause of death worldwide. However, X-ray coronary angiography is often met with limitations associated with the lack of uniformity with illumination, as well as the presence of artifacts and noise. The rapid development and promising prospects of object detection methods can be leveraged to address this issue. Recent work on stenosis detection employs a two-stage Region Proposal Network trained with bounding-box annotations for object classification and bounding-box regression. However, this densely tiling process is time-consuming and reduces diagnostic efficiency. This work proposes a lightweight one-stage detector using a redesigned RetinaNet with optimized anchor configurations to meet particular challenges that exist in medical imaging. Additionally, image augmentation techniques were utilized to improve the medical dataset. This one-stage stenosis detector achieved 0.735 precision and 0.721 recall on the inference of stenosis regions of interest, significantly outperforming the state-of-the-art one-stage approach by 30%. The contribution of this work opens the path to a fully automated diagnostic tool for Coronary Artery Disease from X-ray coronary angiograms to better evaluate the prognosis of patients, bringing great clinical value in the field of cardiothoracic surgery.

Index Terms—Object Detection, Segmentation and Categorization, Computer Vision for Medical Robotics, Deep Learning Methods

I. INTRODUCTION

Coronary Artery Disease (CAD), characterized by plaque accumulation [1], is the leading cause of death worldwide, and the most common cause of morbidity and mortality [2]. It accounts for 20% of all deaths in Europe [3] and 30% of all deaths in the US [4] for over 35 years. The main cause of CAD is plaque buildup in the epicardial coronary arteries, which reduces cardiac blood flow and causes a mismatch between myocardial oxygen supply and demand [1]. This plaque buildup causes the inside of the arteries to narrow over time; the narrowing of coronary arteries is termed coronary stenosis.

Despite advancements in novel imaging modalities, Invasive Coronary Angiography (ICA) remains the gold standard diagnostic tool for Coronary Artery Disease, according to The European Society of Cardiology [5]. This procedure involves threading a catheter into the blood vessel and

¹Jennifer Liu is with the School of Engineering, Tufts University, Medford, MA, USA jennifer.liu@tufts.edu

²James K. Miller and Artur Dubrawski are with the Auton Lab, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA mille856@andrew.cmu.edu, awd@andrew.cmu.edu

³Keith Dufendach is with the Department of Cardiothoracic Surgery, University of Pittsburgh Medical Center, Pittsburgh, PA, USA dufendachka@upmc.edu

obtaining dynamic X-ray images. The assessment of CAD requires interpretive expertise to evaluate multiple parameters, including the number of affected major coronary arteries, the location of lesions, and the severity of stenosis. In addition to being labor-intensive and time-consuming, visual assessment of stenosis severity is associated with high intraobserver and interobserver variabilities due to the lack of uniformity with illumination, as well as the presence of artifacts and noise [6]. Studies demonstrate that visual interpretations of angiograms underestimate the degree of underlying disease, which ultimately leads to a delayed diagnosis [7]. This calls for an automated assessment pipeline for Coronary Artery Disease.

Prior works in the field of automated stenosis detection have successfully identified regions with stenosis. A two-stage Regional Proposal Network model is often employed for this detection task, which raises concerns around the efficiency of this approach. While two-stage detectors yield accurate results, they might not be optimal for this specific clinical application. Since coronary angiography is an invasive procedure that involves radiologic exposure and obviating repeated contrast injections, the slow image processing speed in two-stage detectors might be a hindrance to diagnostic efficiency. One-stage detectors (i.e. YOLO, SSD, RetinaNet) can be leveraged to detect and grade stenosis at a much greater speed. YOLOv5 [8] detectors are the most widely used anchor-based one-stage architectures among the one-stage family [9]. In addition to that, a research gap still exists in the overall pipeline for Coronary Artery Disease diagnosis. Minimal interventions have been made to support physicians to determine the right treatment option for the patient. To determine the right treatment option, physicians are required to assess the grade of the stenosis and symptoms to recommend treatment (i.e. Coronary Artery Bypass Graft, Percutaneous Coronary Intervention).

We propose a one-stage detector approach to generate possible stenosis regions of interest in coronary angiography, an integral part of the pipeline towards grading stenosis and diagnosing Coronary Artery Disease (Fig. 1). In particular, this work aims to select the most efficient Convolutional Neural Network (CNN) architectures and explore optimization methods for detecting and grading coronary artery stenosis. By generating images with possible stenosis, further segmentation and classification tasks can be performed to create interpolation of the degrees of vessel obstruction and generate SYNTAX scores to determine accurate treatment options for patients (Fig. 1).

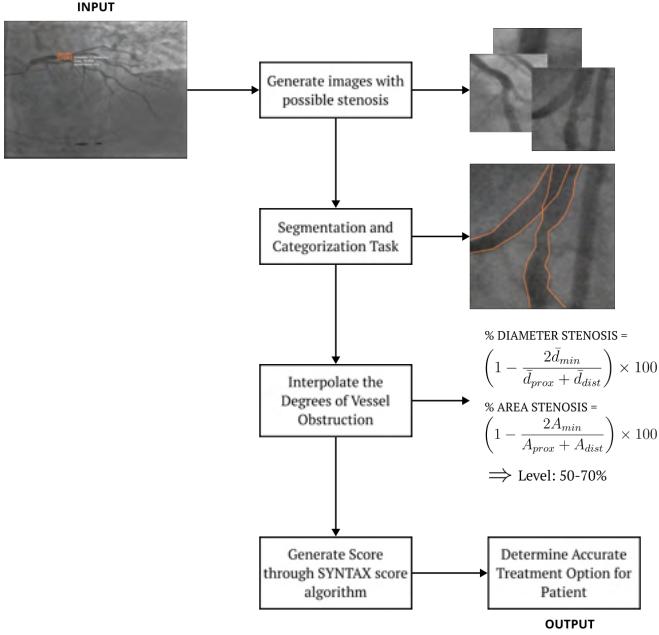


Fig. 1. Proposed End-to-End Pipeline for the Diagnosis of Coronary Artery Disease.

In summary, the contributions of our work are three-fold:

- This work proposed an end-to-end assessment pipeline for Coronary Artery Disease as a diagnostic tool to support physicians.
- This work addressed the importance of diagnostic efficiency by leveraging the power of one-stage detection approaches.
- This work selected the most efficient CNN architecture to construct a model that optimizes speed and accuracy to generate regions of interest containing possible stenosis in Coronary Angiography images

II. RELATED WORKS

This section highlights prior works in stenosis detection, background information about object detection algorithms, and preliminary research on object detection, segmentation, and classification in the wider medical imaging field.

Several approaches for automated assessments of Coronary Artery Disease and stenosis detection have been proposed by different research groups within the medical imaging field. Tools such as centerline extraction, graph-based method, superpixel mapping, and machine or deep learning have been explored, with machine or deep learning being most powerful and have shown great promise in CAD detection due to their performance, tuning flexibility, and optimization [10].

A. Two-stage vs. One-stage Object Detectors

Recent work on stenosis detection employs a two-stage Region Proposal Network, such as Faster R-CNN and R-FCN. Danilov et al. [10] pioneered the study of stenosis detection by analyzing the speed and accuracy trade-off

for detecting single stenosis with specific state-of-the-art CNN architectures. The comparative analysis demonstrated 3 promising neural networks, with Faster-RCNN Inception ResNet V2—a two-stage detector—being the most effective model to detect single-vessel disease. To determine the location of stenosis, Danilov et al. evaluated the distribution of the stenosis coordinates along the vessel in the input images which is then used to estimate the center point of the bounding box around the stenosis.

Despite Faster-RCNN Inception ResNet V2 having the most optimal balance between accuracy and speed, this complicated architecture is not optimized for this application. In Faster-RCNN, the first stage Region Proposal Network proposes candidate object bounding boxes; the second stage consists of feature extraction by ROI Pooling (RoIPool) operation from each candidate box for the following classification and bounding-box regression tasks. Unlike two-stage detectors, one-stage detectors propose predicted boxes from input images directly without region proposal step, thus they are time-efficient and can be used for real-time applications [11]. Towards the pipeline proposed in Fig. 1, an efficient and elegant one-stage approach is favored.

Compared to one-stage detectors, two-stage detectors have an advantage for accuracy performance. By sampling a sparse set of region proposals, two-stage detectors filter out most of the negative proposals, while one-stage detectors directly face all the regions on the image. Lin et al. [12] investigated that the class imbalance problem found in most one-stage detectors contributes to lower accuracy performance. It was found that the extreme foreground-background class imbalance encountered during training of dense detectors is the central cause. By implementing a novel Focal Loss, the vast number of easy negatives can be prevented.

B. Lesion Segmentation

In the wider field of object detection and imaging, novel machine learning models have been developed for lesion segmentation with weak supervision. Weak labels, such as RECIST diameters, are leveraged for segmentation, detection, and classification tasks, as seen in Chu et al. [13] and Zlocha et al. [14]. In particular, Chu et al. proposed a joint classification localization network by clustering input patches containing lesion. Segmented lesion would be used to monitor lesion shape, and may prove crucial to identifying any abnormalities.

Chu et al.'s [13] approach, including many others, show the potential of weak labels for detection, segmentation, and categorization tasks. However, this is not readily available in X-ray coronary angiography and additional preprocessing stages are required; this work aims to explore such possibilities for coronary angiography.

III. METHODOLOGY

This section first formulates the source data, model architecture, and training details for the models. The metrics and

thresholds used to evaluate the performance of models are defined.

A. Source Data

The dataset consists of clinically-obtained coronary angiography sequences of 100 patients at the University of Pittsburgh Medical Center, curated by a Cardiothoracic Surgery expert. A total of 256 images with frame dimensions of 800 by 600 pixels were obtained from these sequences. For each coronary angiography frame containing stenosis, 2D bounding-boxes are labeled to showcase unique stenosis of the corresponding artery and classified based on its degree of stenosis (50-70%, 70-90%, >90%). Examples of labeled source images are shown in Fig. 2.

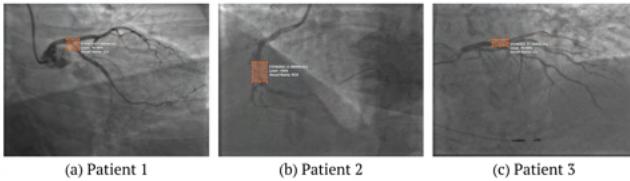


Fig. 2. Data labeling of source images with 2D bounding-boxes of detected stenosis.

B. Model Architecture and Training Details

In order to detect and estimate the position of every visible stenosis in a given frame, a one-stage detector approach is utilized, where the stenosis is the object of interest. We select the backbone of the proposed approach to be a RetinaNet architecture [12], a one-stage detector, highlighted in Fig. 2; the use of a focal loss addresses the common problem of class imbalance in detection tasks in classic one-stage object detection methods, like YOLO and SSD. A top-down architecture with lateral connections is adopted for the Feature Pyramid Network (FPN) to detect objects at different scales and effectively capture information about stenosis of varying sizes [15].

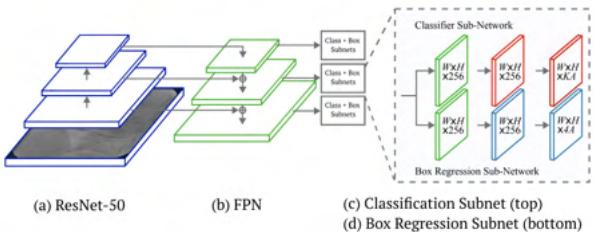


Fig. 3. RetinaNet architecture with (a) ResNet-50 backbone and (b) Feature Pyramid Network as feature extractor to (c) classify the existence probability and (d) regress the bounding box coordinates of stenotic lesions.

Classification and regression sub-networks are responsible for classifying a bounding box and regressing the estimated coordinates. Anchor configurations are crucial for the performance of the detector, and it was found that the default anchor sizes $\{32, 64, 128, 256, 512\}$, aspect ratios $\{1:2, 1:1, 2:1\}$, and scales $\{2^{\frac{0}{3}}, 2^{\frac{1}{3}}, 2^{\frac{2}{3}}\}$ turn out to be ineffective for medical images [12]. We implemented anchor optimization methods proposed by Zlocha et al. that utilize a differential evolution search algorithm to optimize ratios and scales of anchors. Anchor aspect ratios of $\{0.286, 0.5, 1.0, 2.0, 3.5\}$ were obtained to fit the epicardial coronary arteries in different directions and 0.5 offset (half stride) between the center of the first anchor and the top-left corner of the image; anchor scales and sizes remain the same. Optimized anchor configurations were then used to train the detector. For each 800 by 600 pixel image, the α -balanced Focal Loss function with cross-entropy was used with $\alpha = 0.25$ and $\gamma = 2$ to address issues with class imbalance [12].

The dataset is split into 80% for training and 20% for testing. In order to address challenges with the lack of public Invasive Coronary Angiography (ICA) datasets, lightweight preprocessing and image augmentation techniques were implemented to enlarge the size and improve the diversity of the dataset during training. Coronary angiography images were flipped in horizontal directions with 50% chance; corresponding bounding boxes were generated based on augmentation steps. As a result, overfitting is reduced which improves generalization. We employ the Adam optimizer with a learning rate of 10^{-4} . All tests and computations were performed with PyTorch 1.9.0 and Python 3.9.5, equipped with NVIDIA GeForce RTX 2080 Ti.

C. Evaluation Metrics

At inference, a non-maximum suppression layer was implemented to select one entity (bounding box with stenosis) out of many overlapping entities. Additionally, our model is configured to output a maximum of 5 bounding boxes at inference time.

Precision and recall evaluation were utilized to evaluate the accuracy of the detection model; a detection is considered a true positive if the intersection-over-union (IoU) and confidence score are both greater than 0.1. Precision and recall scores were generated per frame based on the maximum IoU score for a detection. The IoU threshold was optimized with the aid of receiver operating characteristics (ROC) curves. In addition to that, the Dice similarity coefficient and inference time per image were calculated in order to quantify the overall performance of the model. For this work, we consider our baseline to be YOLOv5 [8], a state-of-the-art one-stage detector, which was trained with the same configurations. During performance evaluation, the amount of false positive detections was also monitored; the number of false positives is important in clinical applications as they may result in choosing an unsuitable treatment option.

IV. RESULTS

To evaluate the performance of proposed model, this section summarizes the performances of models tested through the chosen evaluation metrics and visual results for stenosis detection.

A. Overall Performance

The best performance of the four models we tested are summarized in Table I. When comparing precision, all of the RetinaNet models outperform the baseline model. However, the baseline model had a greater recall score when compared to the original RetinaNet. The model with the best accuracy performance is the improved RetinaNet with optimized anchors and trained with an improved dataset, obtaining a 0.735 precision, 0.721 recall, and 0.728 Dice coefficient.

TABLE I

COMPARISON OF STENOSIS DETECTION PERFORMANCE IN BASELINE AND RETINANET MODELS

Model	Precision	Recall	Dice Score	Inference Time
Baseline (YOLOv5)	0.3750	0.4030	0.3885	10 ms
Original RetinaNet	0.3815	0.3494	0.3648	35 ms
+ Anchor Optimization	0.5122	0.5060	0.5091	35 ms
+ Image Augmentation	0.7349	0.7211	0.7279	41 ms

Despite its high accuracy, the RetinaNet model with optimized anchors and trained with an improved dataset had the highest amount of variation, with a standard deviation of 0.3. On the other hand, the standard deviation of the baseline model was 0.05. The baseline model performed the fastest, with an inference time of 10 ms per frame whilst the best model had an inference time was 41 ms per frame.

When comparing our model with the two-stage detector with the most optimal balance between accuracy and speed (Faster-RCNN Inception ResNet V2) [10], the performance accuracy gap is still significant. Our model's Dice coefficient is lower than Faster-RCNN Inception ResNet by approximately 0.2. Alternatively, our model demonstrated superior inference time compared to Faster-RCNN Inception ResNet by approximately half time.

Remark 1: Our one-stage stenosis detector achieved 0.728 Dice score on the determination of stenosis regions of interest, significantly outperforming the state-of-the-art one-stage approach by 30%. When comparing with the most optimal two-stage detector, our model had a superior inference time but trailed the accuracy of a two-stage approach.

B. Visualization of Inferences

Fig. 4 highlights visual results of inferences made by the best model on test images, with true positive cases illustrated in the first column and false positive cases illustrated in the second column. The model not only was able to detect ground truth labels, but also stenosis with grades below 50% as seen in some instances of false positive detections (Fig. 1b). This information could be valuable for physicians to easily track stenotic lesion over time. At the same time, this might be detrimental as it may result in choosing an unsuitable treatment option.

When investigating inferences made by different models, our RetinaNet models were able to distinguish coronary arteries from the background well due to Focal Loss. On the

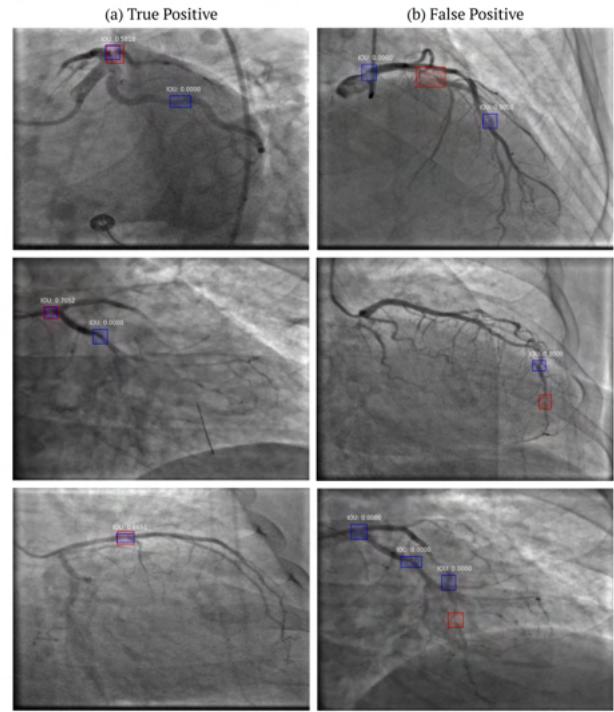


Fig. 4. Visualizations for stenosis detection using best RetinaNet model with Anchor Optimization and Image Augmentation. (a) First column illustrates true positive cases. (b) Second column illustrates false positive cases. Red boxes are ground truth, blue boxes are predicted stenosis with IoU score labeled.

other hand, the baseline model detected more background regions as stenosis due to the lack of Focal Loss.

Remark 2: The best RetinaNet model was not only able to detect ground truth labels, but also stenosis with grades below 50%. All RetinaNet models were able to distinguish coronary arteries from the background, regarding the background regions as easy negatives.

V. CONCLUSION AND FUTURE WORK

This work presents a one-stage approach to generate stenosis regions of interest in X-Ray Coronary Angiography using an improved RetinaNet, outperforming the state-of-the-art baseline by a significant margin. The contribution of this work is shown to be significant for the proposed pipeline for diagnosing Coronary Artery Disease. In particular, this work shows the advantage of one-stage approaches for certain applications in medical imaging. While our approach is only one of many potential ways towards stenosis detection, we hope that our exploration into one-stage detectors for real-time medical imaging tools opens the path to a fully automated diagnostic tool for Coronary Artery Disease to better evaluate the prognosis of patients, bringing great clinical value in the field of cardiothoracic surgery.

For further work, a few adjustments could be made to further improve the performance of our model. We observed large variability in accuracy performance on the best model

compared to the baseline. This large variability could be associated with the amount of noise in the image. A number of image enhancement methods have been proposed by different research groups using tools such as applying a multi-scale top-hat transform [16] to enhance the contrast of input angiogram images. Another improvement is integrating attention mechanism into the feature pyramids in the RetinaNet architecture. A recent attention gate (AG) learns to focus on target structures by producing an attention map, benefiting small, varying structures [17]. This would be useful in improving bounding box detections as stenosis only represents small proportions of the coronary angiography frame.

ACKNOWLEDGMENT

This material is based upon work supported by Carnegie Mellon University's Center for Machine Learning for Healthcare. Thank you to Carnegie Mellon University, Robotics Institute for this research opportunity; A special thanks to the Auton Lab and my mentors—Dr. Kyle Miller, Dr. Keith Dufendach, and Dr. Artur Dubrawski—for their mentorship and guidance throughout this research journey.

REFERENCES

- [1] R. V. Jensen, M. V. Hjortbak, and H. E. Bøtker, "Ischemic heart disease: An update," *Semin Nucl Med*, vol. 50, no. 3, pp. 195–207, 2020, jensen, Rebekka Vibjerg Hjortbak, Marie Vognstoft Bøtker, Hans Erik 2020/4/15. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pubmed/32284106>
- [2] A. Aminorroaya, M. Yoosefi, N. Rezaei, M. Shabani, E. Mohammadi, N. Fattahi, S. Azadnajafabad, M. Nasserinejad, S. Naderimagh, N. Ahmadi, H. Ebrahimi, M. Mirbolouk, M. J. Blaha, B. Larijani, and F. Farzadfar, "Global, regional, and national quality of care of ischaemic heart disease from 1990 to 2017: a systematic analysis for the global burden of disease study 2017," *Eur J Prev Cardiol*, 2021, aminorroaya, Arya Yoosefi, Moein Rezaei, Negar Shabani, Mahsima Mohammadi, Esmaeil Fattahi, Nima Azadnajafabad, Sina Nasserinejad, Maryam Rezaei, Nazila Naderimagh, Shohreh Ahmadi, Naser Ebrahimi, Hooman Mirbolouk, Mohammadhassan Blaha, Michael J Larijani, Bagher Farzadfar, Farshad 2021/5/28. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pubmed/34041535>
- [3] N. Townsend, L. Wilson, P. Bhatnagar, K. Wickramasinghe, M. Rayner, and M. Nichols, "Cardiovascular disease in europe: epidemiological update 2016," *Eur Heart J*, vol. 37, no. 42, pp. 3232–3245, 2016, townsend, Nick Wilson, Lauren Bhatnagar, Prachi Wickramasinghe, Kremlin Rayner, Mike Nichols, Melanie 2016/8/16. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC523477>
- [4] E. J. Benjamin, S. S. Virani, C. W. Callaway, A. M. Chamberlain, A. R. Chang, S. Cheng, S. E. Chiuve, M. Cushman, F. N. Delling, R. Deo, S. D. de Ferranti, J. F. Ferguson, M. Fornage, C. Gillespie, C. R. Isasi, M. C. Jiménez, L. C. Jordan, S. E. Judd, D. Lackland, J. H. Lichtman, L. Lisabeth, S. Liu, C. T. Longenecker, P. L. Lutsey, J. S. Mackey, D. B. Matchar, K. Matsushita, M. E. Mussolino, K. Nasir, M. O'Flaherty, L. P. Palaniappan, A. Pandey, D. K. Pandey, M. J. Reeves, M. D. Ritchey, C. J. Rodriguez, G. A. Roth, W. D. Rosamond, U. K. A. Sampson, G. M. Satou, S. H. Shah, N. L. Spartano, D. L. Tirschwell, C. W. Tsao, J. H. Voeks, J. Z. Willey, J. T. Wilkins, J. H. Wu, H. M. Alger, S. S. Wong, P. Muntner, A. H. A. C. o. E. Subcommittee, P. S. Committee, and S. Statistics, "Heart disease and stroke statistics-2018 update: A report from the american heart association," *Circulation*, vol. 137, no. 12, pp. e67–e492, 2018, benjamin, Emelia J Virani, Salim S Callaway, Clifton W Chamberlain, Alanna M Chang, Alexander R Cheng, Susan Chiuve, Stephanie E Cushman, Mary Delling, Francesca N Deo, Rajat de Ferranti, Sarah D Ferguson, Jane F Fornage, Myriam Gillespie, Cathleen Isasi, Carmen R Jiménez, Monik C Jordan, Lori Chaffin Judd, Suzanne E Lackland, Daniel
- [5] A. Saraste and J. Knuuti, "Esc 2019 guidelines for the diagnosis and management of chronic coronary syndromes : Recommendations for cardiovascular imaging," *Herz*, vol. 45, no. 5, pp. 409–420, 2020, saraste, Antti Knuuti, Juhani 2020/5/21. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC72430520>
- [6] T. A. DeRouen, J. A. Murray, and W. Owen, "Variability in the analysis of coronary arteriograms," *Circulation*, vol. 55, no. 2, pp. 324–8, 1977, deRouen, T A Murray, J A Owen, W 1977/2/1. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC832349>
- [7] J. R. Burton and G. M. FitzGibbon, "Delayed diagnosis of symptomatic coronary artery disease in young men," *Can Med Assoc J*, vol. 106, no. 1, pp. 23–6, 1972, burton, J R FitzGibbon, G M 1972/1/8. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1350032>
- [8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Conference Proceedings, pp. 779–788.
- [9] D. Feng, C. Haase-Schuetz, L. Rosenbaum, H. Hertlein, C. Gläser, F. Timm, W. Wiesbeck, and K. Dietmayer, *Deep Multi-modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges*, 2019.
- [10] V. V. Danilov, K. Y. Klyshnikov, O. M. Gerget, A. G. Kutikhin, V. I. Ganyukov, A. F. Frangi, and E. A. Ovcharenko, "Real-time coronary artery stenosis detection based on modern neural networks," *Sci Rep*, vol. 11, no. 1, p. 7582, 2021, danilov, Viacheslav V Klyshnikov, Kirill Yu Gerget, Olga M Kutikhin, Anton G Ganyukov, Vladimir I Frangi, Alejandro F Ovcharenko, Evgeny A 2021/4/9. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8382165>
- [11] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, "A survey of deep learning-based object detection," *IEEE access*, vol. 7, pp. 128837–128868, 2019.
- [12] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, Conference Proceedings, pp. 2980–2988.
- [13] T. Chu, X. Li, H. V. Vo, R. M. Summers, and E. Sizikova, "Improving weakly supervised lesion segmentation using multi-task learning," in *Medical Imaging with Deep Learning*, Conference Proceedings.
- [14] M. Zlocha, Q. Dou, and B. Glocker, "Improving retinanet for ct lesion detection with dense masks from weak recast labels," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, Conference Proceedings, pp. 402–410.
- [15] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Conference Proceedings, pp. 2117–2125.
- [16] E. Nasr-Esfahani, N. Karimi, M. H. Jafari, S. M. R. Sorourshmehr, S. Samavi, B. K. Nallamothu, and K. Najarian, "Segmentation of vessels in angiograms using convolutional neural networks," *Biomedical Signal Processing and Control*, vol. 40, pp. 240–251, 2018. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1746809417302215>
- [17] J. Schlemper, O. Oktay, M. Schaap, M. Heinrich, B. Kainz, B. Glocker, and D. Rueckert, "Attention gated networks: Learning to leverage salient regions in medical images," *Med Image Anal*, vol. 53, pp. 197–207, 2019, schlemper, Jo Oktay, Ozan Schaap, Michiel Heinrich, Mattias Kainz, Bernhard Glocker, Ben Rueckert, Daniel 2019/2/26. [Online]. Available: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC68802813>

Unsupervised Visual to Thermal Image Translation and Registration

Xinhang Liu¹, Huai Yu² and Sebastian Scherer²

Abstract—In this work, we focus on the topic of infrared (IR) and visual (RGB) image registration, which is a fundamental task for IR-RGB fusion and IR-RGB visual odometry. Especially it enables the early feature fusion for day-night visual odometry. Different from traditional methods based on sparse feature point extraction and matching, we develop a multimodal optical flow estimation framework, which mainly relies on the structure consistency between two modalities to establish dense correspondences. Then we come up with a network structure to predict IR-RGB optical flow. This architecture combines optical flow estimation with multimodal image-to-image translation and can be trained in either a supervised or an unsupervised manner. In order to take advantage of RGB and infrared images’ complementary description of a scene, we use autoencoders to represent the two images as interactive latent codes. The results demonstrate the effectiveness of the proposed IR-RGB network.

Index Terms—Deep Learning for Visual Perception, Image-to-Image translation, IR-RGB registration, optical flow

I. INTRODUCTION

Visual to infrared image registration is a fundamental problem for high-level tasks, such as IR-RGB image fusion [1], fused object detection [2] and state estimation [3]. Due to the different imaging bandwidths of the two sensors, a precise registration and fusion between IR and visual image can greatly enhance the captured information in one single image. The registration of these two modal 2D data is to establish the point correspondences between pixels and then geometric relationship, such as affine transformation and non-rigid optical flow, can be estimated. Despite the current progress in obtain sparse point correspondences [4] and image fusion [1], the dense cross-modal optical flow estimation is still a great challenge. Also there is not a data driven strategy to learning the cross-modal dense correspondences.

Theoretically, IR and RGB cameras have different imaging mechanism. Infrared camera captures information in the longwave infrared wavelength ($8\text{--}15\mu\text{m}$), which is sensitive to the temperature of surrounding environments and the material of reflection objects. While visual camera captures more abundant information in a lower wavelength ($400\text{--}700\text{nm}$), which is similar to the wavelength of human eye’s view. This difference makes the registration of IR to RGB images a big challenge. Due to the geometric edges can be captured by both IR images and visual images, most current

¹Xinhang Liu is with School of Information Science and Technology, ShanghaiTech University, Shanghai, China liuxh2@shanghaitech.edu.cn

²Huai Yu, Sebastian Scherer are with Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA {huaiyu, basti}@andrew.cmu.edu

IR-RGB registration methods are based on this characteristics to find the geometric consistency. Simultaneously, the accurate affine transformation or dense optical flow can be estimated by maximize this consistency. For example, RIFT [4] uses frequency domain to remove the texture difference and encode the geometric information as a descriptor to get sparse point correspondences, then a 3×3 homography matrix can be estimated. However, these kinds of traditional methods mainly reply on the sparse point feature extraction and local texture encoding, which is time-consuming and cannot be used for online visual odometry.

Intuitively, one may consider handling this problem with current deep learning based optical flow estimation methods [5], [6]. However, in RGB-thermal optical flow, the two frames of a pair of input images are in different modalities, while these methods deal with two input images in a same modality. What’s more, these methods only work well in a supervised setting, which means we need training data with ground truth optical flow. However, ground truth optical flow is hardly accessible in our case.

Since the setup of IR and RGB cameras is usually rigidly bundled and synchronized for robotic applications, the dense correspondence is important for an accurate state estimation and image fusion. However, the cross-modal IR-RGB dense correspondence estimation is even more challenging than the homography estimation with sparse point matching. Inspired by the SOTA image-to-image translation framework, we propose to combine the IR-RGB flow estimation and IR-RGB image translation in a novel unsupervised network architecture (as shown in Fig. 1). We first present an encoder-decoder based network to translate between two modalities, then the optical flow estimation can be conducted on the same domain in both RGB branch and IR branch. This combination of cross-modal image translation and flow estimation can benefit from each other. By translating the image style and preserving geometric consistency, IR-RGB image translation can decrease the modality gap between IR and RGB images, thus make flow estimation easier to learn.

II. RELATED WORK

A. GAN-based image-to-image translation

The GAN framework [7] has achieved impressive results in image generation and has also been exploited in image-to-image translation. A conditional generative model [8], which learns translations between domains and applies to the context of source images a target appearance learned from a dataset can align the distribution of translated images with real images in the target domain.

The cycle consistency loss [9], [10], which enforces that if we translate an image to the target domain and back, we should obtain the original image, is a frequently-used constraint used in i2i networks. Another important progress is the assumption of a shared latent space such that corresponding images in two domains are mapped to the same latent code [11], which was further extended by [12].

B. Optical flow estimation

Optical flow describes a dense pixel-wise correspondence between two images, specifying for each pixel in the first image, where that pixel is in the second image. Classical methods formulate optical flow estimation as an optimization problem.

They infer, for a given image pair, a flow field that maximizes smoothness and the similarity of matched pixels. Recent supervised learning approaches instead train deep neural networks to estimate optical flow from examples of ground-truth annotated image pairs [5], [6].

Unsupervised approaches appeared after supervised methods and showed that even without labels, deep learning can greatly outperform classical flow methods [13]. A recent study performed an extensive comparison of the many proposed advances in unsupervised flow estimation and amalgamated these different works into a state of the art method called UFlow [14], [15].

C. Multi-modal image matching

Recently, the multi-modal image matching task has drawn increasingly more attention, and several algorithms have been proposed. For example, local self-similarity descriptor (LSS) [16], partial intensity invariant feature descriptor (PI-IFD) [17], distinctive order-based self-similarity descriptor (DOBSS) [18], ARRSI [19], histogram of orientated phase congruency (HOPC) [20], and phase congruency structural descriptor (PCSD) [21].

RIFT [4] used phase congruency (PC) for feature point detection proposed a MIM measure for feature description¹. Different from these previous work, where feature points are extracted and matched and image registration is done in a such a sparse way, we attempt to formulate this task in a dense manner. We look for the correspondence of each pixel of one image in the other.

III. METHODOLOGY

A. Disentanglement representation of RGB and IR images

Let (x_A^1, x_B^2) be a pair of images, where $x_A^1 \in \mathcal{X}_A$ is an image of frame 1 from RGB domain, and $x_B^2 \in \mathcal{X}_B$ is an image of frame 2 from IR domain. In our setting, we are given samples drawn from the distributions $p(x_A^1, x_B^2)$, without access to the joint distribution $(x_A^1, x_A^2, x_B^1, x_B^2)$. Our goal is to estimate the conditional $p(x_B^2 | x_A^1, x_B^1)$ with a learned model.

To tackle this problem, we make a partially shared latent space assumption like [12]. Specifically, on one hand, we

¹We reimplemented RIFT in C++, with code available at <https://github.com/DarlingHang/RIFT>

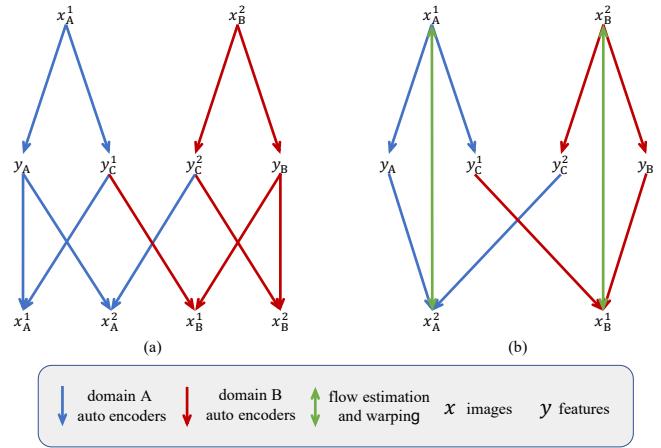


Fig. 1. Our assumptions on images. (a) The input RGB image and IR image can be mapped into a partially shared latent space. These latent codes can be used to translate the input into the other modality, with the geometry remained and the appearance corresponding to the other input. They should also be able to reconstruct the input images. (b) On top of the connection of images through latent space, the two images within a same modality should also be warped toward each other using an estimated optical flow.

assume that each RGB image $x_A \in \mathcal{X}_A$ is generated from a content latent code $y_C \in \mathcal{Y}_C$ that is shared by both domains, and a color latent code $y_A \in \mathcal{Y}_A$ that is specific to the RGB domain. On the other, we assume that each IR image $x_B \in \mathcal{X}_B$ is generated from a content latent code $y_C \in \mathcal{Y}_C$ and a thermal latent code $y_B \in \mathcal{Y}_B$ that is specific to the RGB domain.

In other words, a pair of corresponding images $(x_A^1, x_A^2, x_B^1, x_B^2)$ from the joint distribution is generated by

$$\begin{aligned} x_A^1 &= G_A^*(y_C^1, y_A), \\ x_A^2 &= G_A^*(y_C^2, y_A), \\ x_B^1 &= G_B^*(y_C^1, y_B), \\ x_B^2 &= G_B^*(y_C^2, y_B), \end{aligned} \quad (1)$$

where y_C^1, y_C^2, y_A, y_B are from some prior distributions and G_A^*, G_B^* are the underlying generators. We further assume that G_A^* and G_B^* are deterministic functions and have their inverse encoders $E_A^* = (G_A^*)^{-1}$ and $E_B^* = (G_B^*)^{-1}$. Our goal is to learn the underlying generator and encoder functions with neural networks.

Fig. 1(a) shows an overview of our encoder-decoder architecture and its learning process. Our translation model consists of an encoder E_A (E_B) and a decoder G_A (G_B) for RGB domain X_A (IR domain X_B).

B. Optical flow estimation module

We found that the encoder-decoder architecture described above can generate reasonable pair, but there can be artifacts in details. This is because the encoder-decoder architecture introduced so far is not sufficient for the extraction and utilization of the local information of the picture. So we propose to use the prediction of the optical flow between two pictures to improve the effect of the model.

Given a pair of a pair of corresponding images (x_A^1, x_B^2) , we want to estimate the flow field $V_1 \in \mathbb{R}^{H \times W \times 2}$, which

Fig. 2. Qualitative results of our methods. Input images, translated images by encoder-decoder, estimated optical flows and warped images via the estimated flows are displayed.

for each pixel in x_A^1 indicates the offset of its corresponding pixel in x_B^2 , as well as the flow field $V_2 \in \mathbb{R}^{H \times W \times 2}$, which for each pixel in x_B^2 indicates the offset of its corresponding pixel in x_A^1 .

Since x_A^1 and x_B^2 are in different domains, it is hard to estimate a flow field. We instead estimate the flow field $\hat{V}_1 \in \mathbb{R}^{H \times W \times 2}$, which for each pixel in (x_A^1, x_B^1) indicates the offset of its corresponding pixel in (x_A^2, x_B^2) , as well as the flow field $\hat{V}_2 \in \mathbb{R}^{H \times W \times 2}$, which for each pixel in (x_A^2, x_B^2) indicates the offset of its corresponding pixel in (x_A^1, x_B^1) . Note that x_A^1 and x_B^2 are generated images from the encoder-decoder architecture described above, and (x_A^1, x_B^1) and (x_A^2, x_B^2) are in the same domain.

As is illustrated in Fig. 1, in addition to the encoder-decoder part, we add an optical flow estimation module to our architecture. This flow estimation module is trained synchronously with encoders and decodes. The training process of this module is carried out in an unsupervised manner, using the generated images from encoders and decodes as the targets.

For this module, we want to learn a function f_θ with parameters θ that estimates the flow field for any image pair, such that $\hat{V}_1 = f_\theta((x_A^1, x_B^1); (x_A^2, x_B^2))$. We learn the parameters θ from data of unlabeled image sequences D by minimizing a loss function \mathcal{L} , $\theta^* = \arg \min \mathcal{L}(D; \theta)$.

It should also be noticed that if the ground truth is acces-

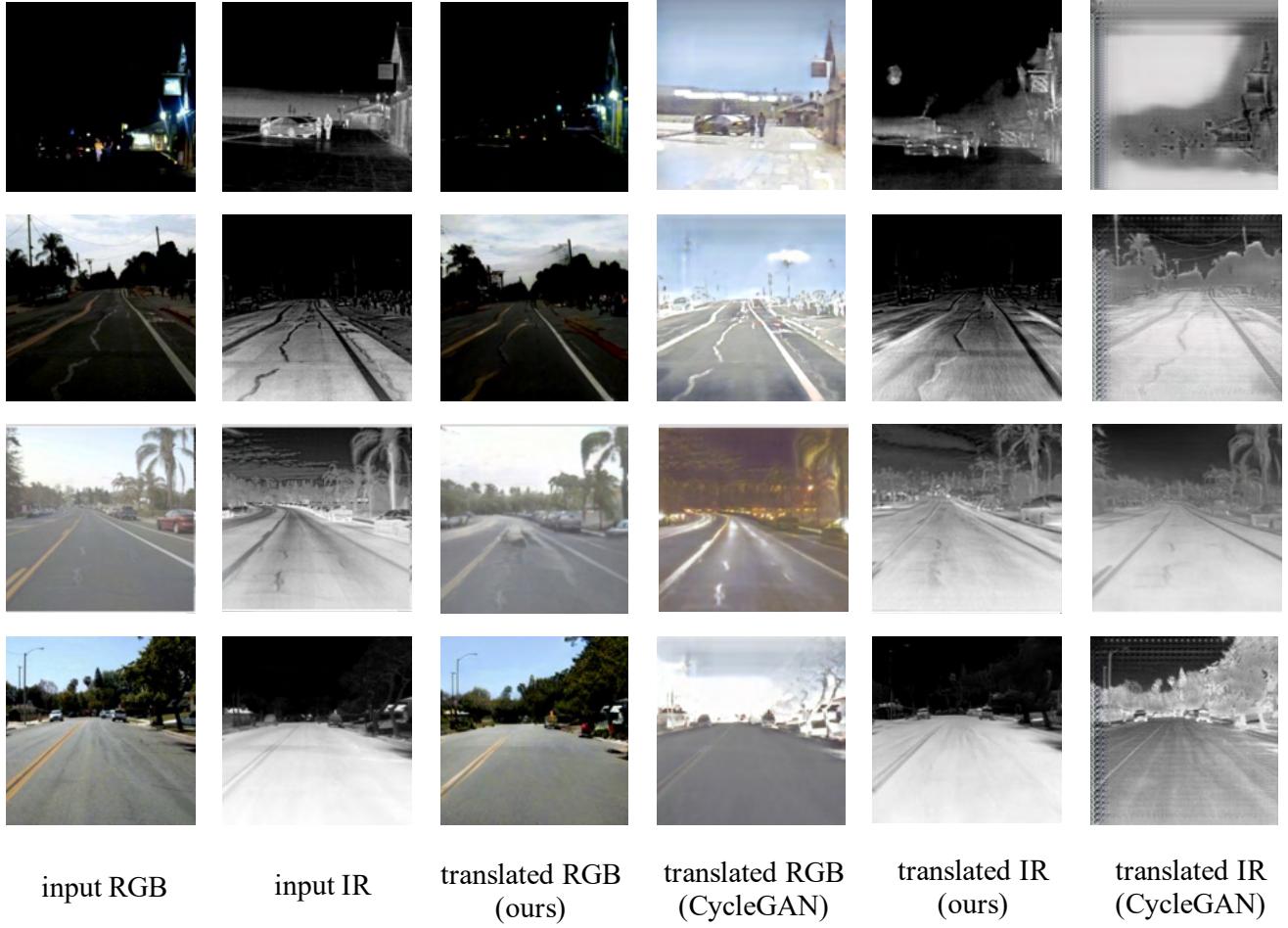


Fig. 3. Translated images by our encoder-decoder architecture, compared with CycleGAN. Not taking advantage of both images' information, CycleGAN can make obvious mistake, generating image whose appearance is inconsistent with the input image of the same modality.

sible, the training procedure of the optical flow estimation module can be converted naturally to a supervised manner.

IV. EXPERIMENTS

In this section, we evaluate our approach with experiments. We first report the implementation details of our approach and the utilized dataset followed by analyzing our results. We further provide the comparison on IR-RGB translation with previous state-of-the-art methods in image-to-image translation.

A. Implementation Details

Our network model is implemented in PyTorch. We run all of our experiments with a single NVidia GeForce RTX3090 GPU. The training time is about 12 hours, with 256×256 cropped input image resolution.

We use the FLIR dataset, which contains around 14K pairs of images. We randomly divided the data into training set and test set according to the ratio of 15:1. This dataset includes scenes of cities and suburbs during the day and night. Among them are the appearance of pedestrians and vehicles.

B. Qualitative Results

Given a pair of input RGB and IR image, our architecture can produce translated images, estimated optical flows as well as warped images. Fig. 3 shows qualitative results of all these components.

Since our encoder-decoder architecture sufficiently utilizes the information from both input images, the translated images have reasonable geometries and appearances. This is further improved by the great performance of our flow module. Without a dependable translation, flow estimation procedure can break down. The flow estimated can be used to warp input images towards each other and in turn refine the encoder-decoder.

C. Comparison of translated images

To evaluate our encoder-decoder architecture, we qualitatively compare translated image by our methods with CycleGAN. 3 shows example results. CycleGAN fail to generate reasonable outputs, either RGB or IR images. Our full model produces images that are realistic and reasonable.

This is because that CycleGAN only takes advantage of one image, instead of complementing two images' descrip-

tion with each other. When translating an RGB image into IR, CycleGAN generated more like a grayscale version of the RGB image. Conversely, when translating an IR image into RGB, Cyclegan paints a grayscale image based on the memory obtained from the training data, and it even turns a picture that should be daytime into night or dusk.

V. CONCLUSIONS

We presented a framework for RGB-IR registration, formulating this problem in a dense manner like optical flow. Our model achieves quality superior to existing image-to-image translation methods and we further use a unsupervised optical flow estimation module to enhance it and preserve more details.

ACKNOWLEDGMENT

This paper was made with the support of Robotics Institute Summer Scholars Program and ShanghaiTech University. Xinhang would like to thank Rachel Burcin, John Dolan and the RISS team for their support during this research project.

REFERENCES

- [1] J. Chen, X. Li, L. Luo, X. Mei, and J. Ma, “Infrared and visible image fusion based on target-enhanced multiscale transform decomposition,” *Information Sciences*, vol. 508, pp. 64–78, 2020.
- [2] J. Heo, S. G. Kong, B. R. Abidi, and M. A. Abidi, “Fusion of visual and thermal signatures with eyeglass removal for robust face recognition,” in *2004 Conference on Computer Vision and Pattern Recognition Workshop*. IEEE, 2004, pp. 122–122.
- [3] M. R. U. Saputra, P. P. de Gusmao, C. X. Lu, Y. Almaliooglu, S. Rosa, C. Chen, J. Wahlinström, W. Wang, A. Markham, and N. Trigoni, “Deepio: A deep thermal-inertial odometry with visual hallucination,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1672–1679, 2020.
- [4] J. Li, Q. Hu, and M. Ai, “Rift: Multi-modal image matching based on radiation-variation insensitive feature transform,” *IEEE Transactions on Image Processing*, vol. 29, pp. 3296–3310, 2019.
- [5] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, “Pwc-net: Cnn’s for optical flow using pyramid, warping, and cost volume,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8934–8943.
- [6] Z. Teed and J. Deng, “Raft: Recurrent all-pairs field transforms for optical flow,” in *European conference on computer vision*. Springer, 2020, pp. 402–419.
- [7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in neural information processing systems*, vol. 27, 2014.
- [8] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.
- [9] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.
- [10] Z. Yi, H. Zhang, P. Tan, and M. Gong, “Dualgan: Unsupervised dual learning for image-to-image translation,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2849–2857.
- [11] M.-Y. Liu, T. Breuel, and J. Kautz, “Unsupervised image-to-image translation networks,” in *Advances in neural information processing systems*, 2017, pp. 700–708.
- [12] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, “Multimodal unsupervised image-to-image translation,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 172–189.
- [13] W. Im, T.-K. Kim, and S.-E. Yoon, “Unsupervised learning of optical flow with deep feature similarity,” in *European Conference on Computer Vision*. Springer, 2020, pp. 172–188.
- [14] J. Janai, F. Guney, A. Ranjan, M. Black, and A. Geiger, “Unsupervised learning of multi-frame optical flow with occlusions,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 690–706.
- [15] R. Jonschkowski, A. Stone, J. T. Barron, A. Gordon, K. Konolige, and A. Angelova, “What matters in unsupervised optical flow,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020, pp. 557–572.
- [16] E. Shechtman and M. Irani, “Matching local self-similarities across images and videos,” in *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.
- [17] J. Chen, J. Tian, N. Lee, J. Zheng, R. T. Smith, and A. F. Laine, “A partial intensity invariant feature descriptor for multimodal retinal image registration,” *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 7, pp. 1707–1718, 2010.
- [18] A. Sedaghat and A. Alizadeh Naeini, “Dem orientation based on local feature correspondence with global dems,” *GIScience & Remote Sensing*, vol. 55, no. 1, pp. 110–129, 2018.
- [19] A. Wong and D. A. Clausi, “Arrsi: Automatic registration of remote-sensing images,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 5, pp. 1483–1493, 2007.
- [20] Y. Ye, J. Shan, L. Bruzzone, and L. Shen, “Robust registration of multimodal remote sensing images based on structural similarity,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 5, pp. 2941–2958, 2017.
- [21] J. Fan, Y. Wu, M. Li, W. Liang, and Y. Cao, “Sar and optical image registration using nonlinear diffusion and phase congruency structural descriptor,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 9, pp. 5368–5379, 2018.

Gander: A Comprehensive Machine Learning Synthesized Media Evaluation Platform & Style-CLIPDraw: A Style-Infused Text-to-Drawing Synthesis Method

Zhixuan Liu¹, Peter Schaldenbrand² and Jean Oh³

Abstract—Machine learning competition websites allow data scientists and engineers to explore and build models in a web-based data-science environment to advance the state of the art on regression and classification tasks. Commonly used data science competition frameworks lack attention to synthesized media such as texts, images, video, or audio. In this work, we introduce a web platform, Gander, which hosts machine learning media synthesis competitions. Gander also aims to evaluate the generated media comprehensively and transparently by implementing the existing code methods for quantitative measurements as well as implementing novel crowdsourcing methods for qualitative evaluations. We also present Style-CLIPDraw, an algorithm that synthesizes drawings based on natural language input and a given style image. Without requiring any training, Style-CLIPDraw takes advantage of a pre-trained VGG16 and a pre-trained CLIP language-image encoder for extracting features and measuring similarities. The performance of generated drawings will be further evaluated by Gander.

Index Terms—Art and Entertainment Robotics

I. INTRODUCTION

In the field of machine learning (ML), competition websites such as Kaggle, AICrowd, and DrivenData host state-of-the-art machine learning tasks and encourage people to solve data science challenges. These competition websites provide practical learning experiences for data scientists and have also led to great innovation in the ML area.

Most competitions hosted by the existing ML platforms are regression and classification tasks, such as value prediction and object detection. With the improvement of computing hardware and the widespread use of deep learning, especially the generative adversarial networks (GANs) by Goodfellow *et al.* [1], media generation, such as image and audio synthesis is a new trend in the ML area. Media generation ML research has experienced a boom, and existing competition websites are slow to adopt the ability to support media synthesis competitions.

Compared with other outputs, media are abstract and complex to evaluate. Some features of media outputs such as aesthetic beauty or novelty are hard to capture by using existing evaluation code metrics. Although there exist some metrics which evaluate the performance of generated media

outputs, they all have their limitations. One metric is only able to capture certain characteristics of generated outputs, and in most cases, the score of the media outputs using one single code metric does not correlate with humans' objectives. In fact, there hasn't been a universal standard in the evaluation of generated media outputs yet, so our goal is to make the evaluation of media outputs comprehensive and easy for users to perform.

We introduce the website Gander, a platform that hosts media generation tasks and is tailored to evaluate media outputs. By surveying the code metrics used in recent media synthesis papers and having conversations with potential users, we select some automatic metrics for operating quantitative measurements as well as using crowdsourcing methods for qualitative analysis.

To test the Gander system, we introduce a novel media generated algorithm to serve as a test case. We were inspired by recent dramatic progress in the linking between text and image such as the CLIP [2] model and the CLIPDraw [3]. We present Style-CLIPDraw, an algorithm that synthesizes novel style-transferred drawing-like images based on a natural language input and a style image input.

Style-CLIPDraw has two main components: one is the text-to-image synthesis, the other is the style transferring from the style image to the content image. This algorithm does not require any training; rather it takes advantage of two pre-trained models. A pre-trained CLIP model is used to measure the similarity between the natural language input and the generated image; a pre-trained VGG16 trained on the ImageNet is used for capturing the style features and then calculating the difference between the style image and the generated image. Similar to CLIPDraw [3], Style-CLIPDraw optimizes the vector strokes rather than pixel images. Therefore, the images Style-CLIPDraw synthesized can be style-specific drawing-like images.

In the following paper, section III presents the user interface and some basic functions of the Gander website, section IV introduces the evaluation methods that are used in Gander for media outputs, and section V introduces the Style-CLIPDraw algorithm.

II. RELATED WORK

Generative Adversarial Networks (GANs) have achieved great results in the media synthesis area, such as image generation [1], [4], music synthesis [5], and text synthesis [6]. The main idea of GANs is the adversarial loss.

¹Zhixuan Liu is with the School of Data Science, the Chinese University of Hong Kong, Shenzhen, China liuzhixuan@cuhk.edu.cn

²Peter Schaldenbrand is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA pschalde@cs.cmu.edu

²⁻³Jean Oh is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA jeanoh@cmu.edu

A GAN is composed of a generator and a discriminator, where the generator tries to synthesize media that fool the discriminator; and the discriminator tries to distinguish generated media from real media. In our work, we present Gander to evaluate the performance of GANs by operating on the synthesized media.

Evaluating GANs. Generative models, in particular, generative adversarial networks (GANs) by Goodfellow *et al.* [1] have gained significant attention in recent years and are widely used in media generation tasks. There exist two approaches for evaluating GANs: qualitative evaluations and quantitative measurements, with their own strengths and weaknesses. Qualitative evaluations, such as fooling a person in distinguishing generated media from real ones are somehow the ultimate test for machine learning generated media [7]. However, such measurements are expensive and difficult to perform for large-scale outputs.

Quantitative measurements, although less subjective, sometimes may not correspond to human evaluations of the generated images [7].

To make comprehensive evaluations for generated media outputs, crowdsourcing methods are used in Gander for qualitative evaluations; Gander also embeds several automatic metrics for quantitative measurements. By standardizing the inputs and outputs of the code, Gander makes it easier for users to implement different measuring strategies on their models by simply selecting the metrics they want to use on the web interface.

Text-to-Image Synthesis. Generating an image based on a text description has attracted a variety of proposed solutions. In the most common approaches, conditional GANs are used by taking a language embedding as input [8] and then generate corresponding image outputs. Commonly used datasets for this approach such as MS-COCO [9] contain paired real natural images and their text descriptions. Thus, the images synthesized by GANs approach are photorealistic.

Recently, the DALL-E model [10] has achieved great success in text-to-image synthesis. It eschews the use of GANs; rather it combines a GPT-3 transformer [11] and a VQ-VAE encoding model [12]; then a state-of-the-art text-image matching model CLIP [2] is used to select output images that maximize the semantic consistency. Moreover, CLIP shows potential as an in-the-loop evaluation method for text-to-image synthesis, and in this work, we follow this idea for the text-to-image synthesis.

Style Transfer. Style transfer takes two images—a content image and a style image—and blends them so that the resulting output image retains the core elements of the content image, but appears to be in the style of the style image. Neural Style Transfer introduced by Gatys *et al.* [13] is the state-of-the-art method to transfer the style of images. Style features of input and outputs images are extracted from different layers of a pre-trained neural network and are then represented by the Gram matrix. The style loss between two images is further calculated by the Frobenius norm of two resulting matrices.

In this work, we are inspired by the Style Transfer by

Relaxed Optimal Transport and Self-Similarity (STROTSS) algorithm [14], where style features are extracted from different layers of a pre-trained VGG16 and style loss is calculated using the Relaxed Earth Movers Distance (EMD).

Synthesis Through Optimization. This work, instead of directly learning an image generative network, synthesizes images through evaluation time. This *activation maximization* method is first proposed by Erhan *et al.* [15], where a random image is optimized through backpropagation to increase the activation of certain neurons in the pre-trained network. The activation maximization method produces highly realistic synthesis images while understanding the meaning of neuron activation is a challenge [3]. In this work, we follow the idea of optimizing during evaluation time: differentiable curves receive two signals and then gradient descent is applied to optimized the curves.

III. USER INTERFACE AND FUNCTIONS OF GANDER

Gander is specially designed for researchers who create new machine learning methods in the Creative Artificial Intelligence (AI) area, or artists who use AI for generating new art pieces. Therefore, the design should satisfy the aesthetic beauty and also contain some creative features. Blue, which can enhance people's performance on creative tasks [16] is chosen to be the theme color. In the following subsections, the main components of Gander (current version) will be presented.

A. Login/Sign up

The Login/Sign-up page of the Gander website is shown in Figure 1. The background image is composed of small doodles, and a "Gander" animation is displayed in the center of the page. We think this design matches the art theme of this platform.

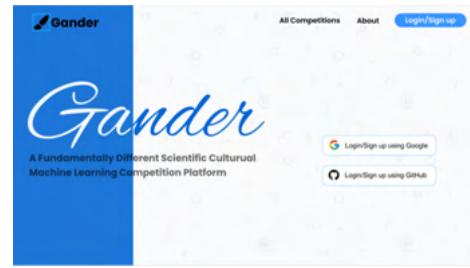


Fig. 1. Login/Sign up page of Gander website

B. Competition Library

Gander hosts many media generation competitions in different genres, including text, image, video, and audio generation. Figure 2 shows the competition system of Gander. The "Your Competition" page lists the history of the competition of users. Users can use the selection bar on the left as a filter to find the competitions they have participated in or bookmarked before.

Competition Library, which is shown in 2 (right) is a place where all the competitions hosted in Gander can be found. Users are able to use tags as filters to target certain

competitions, bookmark competitions, and also create their own competitions.

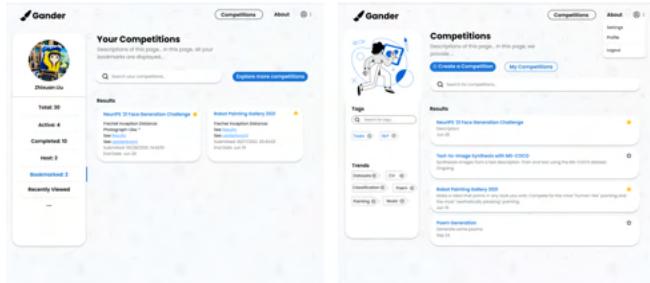


Fig. 2. Competition Library: My Competitions & All Competitions

C. Competition Page

When users enter a competition, some basic information of this competition is displayed in the interface, including the overview description of this competition, data description (shown in Figure 3), and some evaluation methods used to evaluate the performance of the generated outputs (shown in Figure 4). The leaderboard is used to rank the results of individuals using the scores of evaluation metrics (shown in Figure 5).

Moreover, compared to other ML competition platforms, Gander is tailored to media outputs, where human evaluation is considered as the ultimate measurement of these tasks. Therefore, users are invited to judge the performance of generated media (shown in Figure 6). The score of human evaluations will also be displayed on the leaderboard.

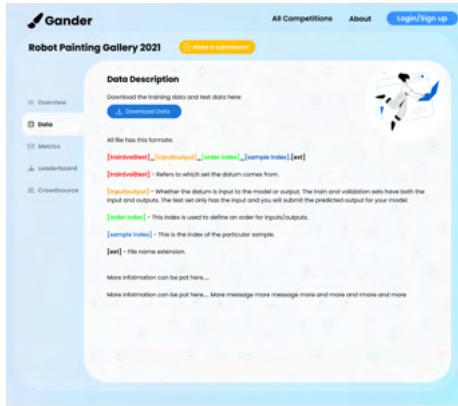


Fig. 3. Competition Page: Data Description

D. Submitted Results

The submitted results page for a competition is shown in Figure 7. The ranks of this submission according to its scores on different evaluation methods are shown. Moreover, Gander also displays media outputs like an art gallery, so that users can directly visualize the performance of image outputs, or directly listen to some synthesis music of this submission.

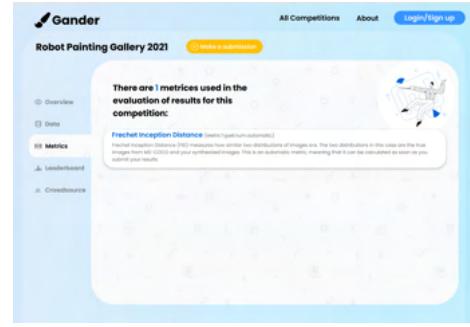


Fig. 4. Competition Page: Metrics used for this competition

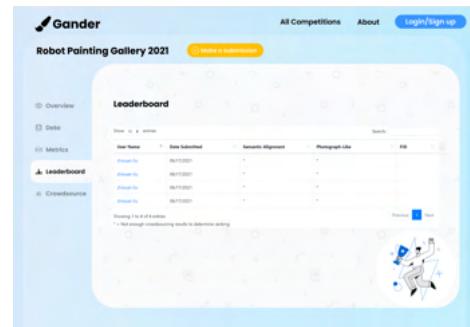


Fig. 5. Competition Page: Leaderboard of this competition

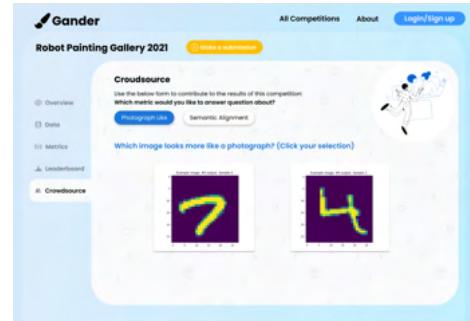


Fig. 6. Competition Page: Metrics used for this competition

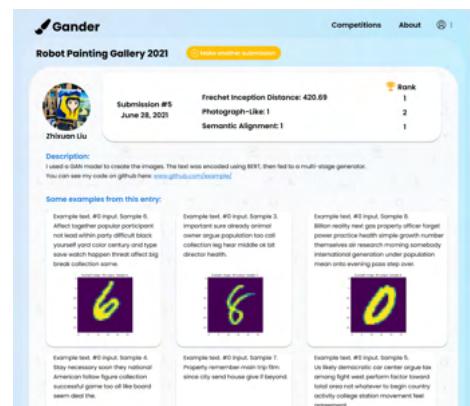


Fig. 7. Submitted results

IV. STATE-OF-THE-ART METHODS FOR EVALUATING GENERATED MEDIA

In this section, we present some state-of-the-art automatic metrics that have been used to evaluate the generated images on the Gander website.

A. Inception Score (IS)

IS *et al.* [17] is one of the most widely adopted score for GAN evaluation (*e.g.* in [18]). Although with some limitations [19], the Inception score shows a reasonable correlation with the quality and diversity of generated images [17]. Therefore, Gander uses IS as one of the automatic metrics to measure the performance of generated images in several tasks, including image synthesis, image inpainting, style transformation, etc.

By using IS, Gander aims to capture two desirable properties of generated images: 1. high classifiable - images contain clear objects and 2. generative algorithm should output a high diversity of images. To capture these two features, the generated images are feed to a pre-trained neural network (the Inception Net [20] on the ImageNet [21]). IS measures the average KL divergence between $p(y|x)$, the conditional label distribution of samples (expected to have lower entropy, because the higher quality of the generated image, the easier it is for samples to be classified by Inception Net [20], and the lower the entropy will be.), and the marginal distribution $p(y)$ of all the samples (expected to have high entropy because it prefers the diversity of generated images). IS can be expressed in the following formula:

$$\exp(\mathbb{E}_x[\mathbb{KL}(p(y|x)||p(y))]) = \exp(H(y) - \mathbb{E}_x[H(y|x)])$$

where $p(y|x)$ is the conditional label distribution for image x predicted using Inception Net [20], and $p(y)$ is the marginal distribution: $p(y) \approx \frac{1}{N} \sum_{n=1}^N p(y|x_n = G(\mathbf{z}_n))$. $H(x)$ is the entropy of x . The higher IS, the better performance of the generative model.

B. Fréchet Inception Distance (FID)

Introduced by Heusel *et al.* [22], FID is used to measure the similarity between two datasets of images. FID performs well in terms of discriminability, robustness and computational efficiency and is consistent with human judgements [7], therefore, it is now widely used in the evaluation of various image generation tasks (*e.g.* in [23] [24]).

Gander calculates FID for the generated data and the real data to measure the similarity of these two datasets statistically. The lower FID, the better performance of the generative model. Gander realizes this function by using a pre-trained Inception Net. Two datasets are embedded into a feature space given by a specific layer of Inception Net. Then, the mean and covariance are estimated for both datasets. FID can be further calculated in the following formula:

$$FID(r, g) = \|\mu_r - \mu_g\|_2^2 + Tr(\sum_r + \sum_g - 2(\sum_r \sum_g)^{\frac{1}{2}})$$

where (μ_r, \sum_r) are the mean and covariance of the real data and (μ_g, \sum_g) are of the generated data.

C. Structural Similarity Index Measure (SSIM) and Peak Signal-to-Noise Ratio (PSNR)

GANs have also been commonly used in the image inpainting tasks (*e.g.* in [25]) and image super-resolution tasks (*e.g.* in [26]), where ground truth images exist and can be directly used to evaluate the performance of generated images. Gander uses SSIM and PSNR to compare the generated results with the corresponding ground truth images.

1) **SSIM**: To be more specific, SSIM compares corresponding pixels and their neighborhoods in the real image (x) and the generated image (y) by capturing three quantities: luminance (I), contrast (C), and structure (S). The numerical expressions of these three characteristics and the SSIM expression are:

$$I(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1}$$

$$C(x, y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}$$

$$S(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3}$$

$$SSIM(x, y) = I(x, y)^\alpha C(x, y)^\beta S(x, y)^\gamma$$

where μ_x, μ_y, σ_x , and σ_x are the mean and standard deviations of pixel intensity in a local image patch centered at x or y , σ_{xy} is the sample correlation coefficient between x and y , and C_1, C_2 , and C_3 are constants that guarantee the numerical stability. The higher the SSIM score, the higher the structural similarity between the two images.

2) **PSNR**: PSNR is another metric Gander uses to measures the peak signal-to-noise ratio between the generated image I and ground truth image K . Higher PSNR (in dB) means better generative quality. PSNR is calculated as:

$$PSNR(I, K) = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right) \\ = 20\log_{10}(MAX_I) - 20\log_{10}(MSE_{I,K})$$

where

$$MSE_{I,K} = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (I(m, n) - K(m, n))^2$$

V. STYLE-CLIPDRAW

In this section, we present Style-CLIPDraw, a method that synthesizes novel style-transferred drawing-like images based on a natural language input and a style image input.

The objective of Style-CLIPDraw is to synthesize a drawing that matches a given description prompt and also has the style of a given style image (some examples in Figure 9). And Figure 8 is the overview of our approach.

Style-CLIPDraw synthesizes drawing-like images. The drawing generated by the Style-CLIPDraw method is composed of a set of differentiable Bézier curves. Style-CLIPDraw controls three features of a Bézier curve: the curve control points, thickness, and the RGBA color vector. Initially, all the features of the curves in the drawings are

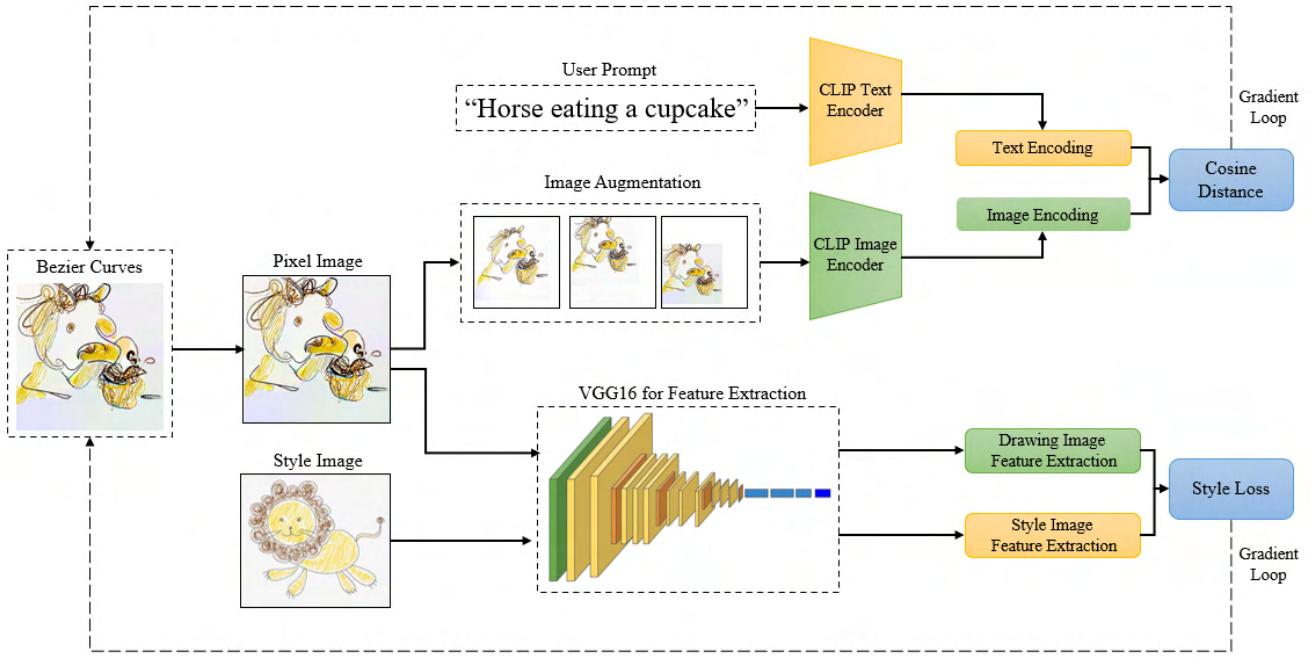


Fig. 8. Summary of our approach: Style-CLIPDraw. Starting from a random set of Bézier curves, the drawing receives two losses: one is the cosine distance between the image encoding and the text prompt encoding; the other is the style loss calculated by the self-similarity of the style image and the drawings. The curves are optimized in the generating process.



Fig. 9. Here are some image synthesis examples using our Style-CLIPDraw. Given the user prompt: "Horse eating a cupcake." and a style image, our algorithm generates a drawing that match these two features.

assigned randomly on the default white background (see Figure 10).

Style-CLIPDraw synthesizes a drawing through optimization. In this process, the number of curves is fixed and the positions of control points, thickness, and RGBA color can be changed during optimization. To optimize a drawing, Style-CLIPDraw receives two signals: one is the content loss which aims to match the user prompt with the synthesis drawing; the other is the style loss which measures the style

distance between the style image and the current drawing.

We calculate the content loss in the following steps. First, the synthesis drawing made up of Bézier curves is converted to a pixel image. Then, the resulting image is duplicated and augmented, which aims to prevent the generation of adversarial examples. Next, a pre-trained CLIP model is used to connect the natural language and the synthesis drawing. The CLIP text encoder maps the natural language input to a 512-length text encoding vector, and the CLIP image encoder

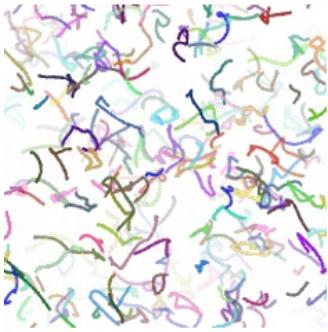


Fig. 10. This figure show the initial image of the drawing where all the Bézier curves are randomly assigned.

maps the given style image to a 512-length image encoding. The content loss is the cosine distance between these two encodings.

Style loss between the synthesis drawing and the style image is calculated in the following steps. Firstly, the curves are converted to a pixel image. Then, both this resulting image and the style image are fed to the VGG16 for feature extraction. We calculate the style loss, which is composed of the relaxed earth movers distance, moment matching loss, and the color matching loss of these two images, following the methods proposed by Kolkink *et al.* [14].

A synthesis drawing is optimized by running evaluation-time gradient descent on both the style loss and content loss. The Style-CLIPDraw algorithm is presented in Algorithm 1.

Algorithm 1 Style-CLIPDraw Algorithm

```

1: Input: Description Phrase desc; Style image styleImg;
   Number of Iterations I; Number of Curves N; Pre-trained
   CLIP; Pre-trained VGG16.
2: Begin:
3: Encode Description Phrase. EnDesc = CLIP-
   textEncoder(desc).
4: Extract Features from the Style Image. FeaStyle =
   VGG16(styleImg).
5: Initialize Random Curves. Curves = RandomCurves().
6: for i=0 to I do
7:   Curves to Pixel image. PixImg = ToImg(Curves).
8:   Augment Image. AugBatch = Augment(PixImg)
9:   Encode Augmented Image. EnImg = CLIP -
   imageEncoder(AugBatch).
10:  Extract Features from the Pixel Image. FeaPix =
    VGG16(PixImg).
11:  Calculate Content Loss for Phrase and Drawing.
    CLoss = -CosSim(EnDesc, EnImg).
12:  Calculate Style Loss for Style Image and Drawing.
    SLoss = styleLoss(FeaStyle, FeaPix).
13:  Total Loss. Loss = CLoss + SLoss.
14:  Backprop. Curves  $\leftarrow$  Minimize(Loss).
15: end for
```

A. Results

In the figure 9, we presents some images synthesized by Style-CLIPDraw. The first row is the given style images, the second row is the synthesis drawings with user prompt “Horse eating a cupcake”, and the third row is the synthesis drawings with the description “Monkey playing the guitar”. In all these sample results, we set the number of curves to be 256, the number of iteration to be 800; for style feature extraction we use layers 9, 10, 12, and 13 of VGG16.

By observing the results of some cases, we conclude some properties of Style-CLIPDraw:

First, The drawings are rendered in a painterly style. This is because Style-CLIPDraw operates on curves rather than pixel images. Therefore, the synthesis images of Style-CLIPDraw look more like human drawings.

Second, style learning of the synthesis drawing is by imitating the colors and strokes from the given style image. This observation further explains a phenomenon: the performance of Style-CLIPDraw is directly related to the given style image. Generally, when style images are real drawings, the synthesis images from Style-CLIPDraw have better performances, because real strokes are easier to imitate by generated curves. However, some concrete features of the style image may also occur in the synthesis drawing in some cases. For example, in the first example with user prompt: “Horse eating a cupcake” in Figure 9, where the dog’s nose in the style image occurs in the synthesis drawing. This may be due to the large iteration times and can be further studied.

Third, the drawings generated by Style-CLIPDraw fit well with the user prompt. This implements that the CLIP model based on the 400 million paired training data set is very powerful. Also, the image augmentation avoids generating adversarial samples, which fool the network but are unrecognizable to humans.

Moreover, when the given style image is a white background color, the objects in the generated picture can be distinguished more clearly. This is because the background color of the initial generated drawing is also white. If the theme color of the style image background is much darker, such as Van Gogh’s “The Starry Night” in Figure 9, curves need to increase their width to cover a large area of the background plate.

VI. CONCLUSION AND FUTURE WORK

Overall, in this paper, we first introduce a web platform Gander, which aims to hosts media generation competitions in different genres and evaluate the generated media outputs comprehensively and transparently. Then, we present some automatic code metrics currently embedded in the Gander websites. Lastly, we introduce Style-CLIPDraw, an algorithm synthesizing novel style-transferred drawing-like images based on a natural language input and a style image input.

For future work, we will continue improving the Gander website and embed more novel evaluation methods for different media outputs. To further improve Style-CLIPDraw, we will find the most suitable weight of two losses in

this algorithm, and then evaluate the performance of Style-CLIPDraw using the Gander website.

ACKNOWLEDGMENT

The authors thank all the Carnegie mellon University Robotics institute summer scholars (RISS) program sponsors, mentors, and partners, especially Rachel Burcin, John Dolan.

REFERENCES

- [1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in neural information processing systems*, vol. 27, 2014.
- [2] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, “Learning transferable visual models from natural language supervision,” *arXiv preprint arXiv:2103.00020*, 2021.
- [3] K. Frans, L. Soros, and O. Witkowski, “Clipdraw: Exploring text-to-drawing synthesis through language-image encoders,” *arXiv preprint arXiv:2106.14843*, 2021.
- [4] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.
- [5] O. Mogren, “C-rnn-gan: Continuous recurrent neural networks with adversarial training,” *arXiv preprint arXiv:1611.09904*, 2016.
- [6] Y. Zhang, Z. Gan, K. Fan, Z. Chen, R. Hieno, D. Shen, and L. Carin, “Adversarial feature matching for text generation,” in *International Conference on Machine Learning*. PMLR, 2017, pp. 4006–4015.
- [7] A. Borji, “Pros and cons of gan evaluation measures,” *Computer Vision and Image Understanding*, vol. 179, pp. 41–65, 2019.
- [8] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, “Generative adversarial text to image synthesis,” in *International Conference on Machine Learning*. PMLR, 2016, pp. 1060–1069.
- [9] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *European conference on computer vision*. Springer, 2014, pp. 740–755.
- [10] A. Ramesh, M. Pavlov, G. Goh, S. Gray, C. Voss, A. Radford, M. Chen, and I. Sutskever, “Zero-shot text-to-image generation,” *arXiv preprint arXiv:2102.12092*, 2021.
- [11] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, “Language models are few-shot learners,” *arXiv preprint arXiv:2005.14165*, 2020.
- [12] A. Razavi, A. van den Oord, and O. Vinyals, “Generating diverse high-fidelity images with vq-vae-2,” in *Advances in neural information processing systems*, 2019, pp. 14 866–14 876.
- [13] L. A. Gatys, A. S. Ecker, and M. Bethge, “Image style transfer using convolutional neural networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2414–2423.
- [14] N. Kolkin, J. Salavon, and G. Shaknarovich, “Style transfer by relaxed optimal transport and self-similarity,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 051–10 060.
- [15] D. Erhan, Y. Bengio, A. Courville, and P. Vincent, “Visualizing higher-layer features of a deep network,” *University of Montreal*, vol. 1341, no. 3, p. 1, 2009.
- [16] Z. Gray, “Creativity through the use of color as an external stimulus,” Ph.D. dissertation, 2010.
- [17] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training gans,” *Advances in neural information processing systems*, vol. 29, pp. 2234–2242, 2016.
- [18] W. Fedus, M. Rosca, B. Lakshminarayanan, A. M. Dai, S. Mohamed, and I. Goodfellow, “Many paths to equilibrium: Gans do not need to decrease a divergence at every step,” *arXiv preprint arXiv:1710.08446*, 2017.
- [19] S. Barratt and R. Sharma, “A note on the inception score,” *arXiv preprint arXiv:1801.01973*, 2018.
- [20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.
- [21] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
- [22] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “Gans trained by a two time-scale update rule converge to a local nash equilibrium,” *Advances in neural information processing systems*, vol. 30, 2017.
- [23] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, “Semantic image synthesis with spatially-adaptive normalization,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2337–2346.
- [24] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, “Maskgan: Towards diverse and interactive facial image manipulation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5549–5558.
- [25] Y. Song, C. Yang, Z. Lin, X. Liu, Q. Huang, H. Li, and C.-C. J. Kuo, “Contextual-based image inpainting: Infer, match, and translate,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19.
- [26] Y. Yuan, S. Liu, J. Zhang, Y. Zhang, C. Dong, and L. Lin, “Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 701–710.

Development of a Force Sensing Glove for The Analysis of Dynamic Motions

Molly Loughney¹, Cornelia Bauer² and Nancy S. Pollard³

Abstract—As robots become more integrated into everyday human life, it is important to understand how they interact with their environments, and ways in which they can use these interactions to their benefit. One way of understanding how robots may interact with their environments is to study human environmental interactions and incorporate them into robotic control models. Current research focuses on using data gathered by motion capture or fixed force plates to help inform these models, however these methods are not necessarily accessible or adaptable for studying motion in a variety of environments. A more accessible method for collecting dynamic force data is to use a sensorized glove, however present available models are expensive and lack the force range needed for dynamic movement. Additionally, many of these models are not adaptable to different end effector configurations with the majority being designed for human hands. In this paper we present the development of a low cost, portable, easily adaptable mitten that can be worn by humans or placed on robotic end effectors. This mitten will allow data about the forces experienced during dynamic motion to be more easily attained, thus helping better tune models for robotic control.

Index Terms—Wearable Robotics, Soft Robotics Applications

I. INTRODUCTION

As we expand the capabilities of robots to interact with humans, the use of robotic end effectors in dynamic motion is becoming an important field of study. Humans use their hands in a variety of dynamic movements such as leaning against furniture while navigating cluttered spaces, pushing against walls to quickly change directions or grasping while in motion. Understanding the forces experienced by the hands during these movements is important for aiding the tuning of control models as they can be used to approximate the forces experienced by a robot in similar circumstances. These approximations can then be applied to simulations to test different control models, primarily centroidal dynamics or CDM. CDM approximates the body of a robot as an ellipsoid centered on the center of mass and has been employed in the modelling of legged locomotion [1] and is especially applicable to the use of force measurements. Currently, the forces experienced by humans during dynamic motions can be measured using technology such as motion

¹Molly Loughney is a Robotics Institute Summer Scholar at Carnegie Mellon University and an undergraduate Engineering Science senior at Smith College, Northampton, MA.
mloughney@smith.edu

²Cornelia Bauer is a PhD student at the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.
cornelieb@andrew.cmu.edu

³Dr. Nancy S. Pollard is a professor at the Robotics Institute and Computer Science Department, Carnegie Mellon University, Pittsburgh, PA.
npollard@andrew.cmu.edu

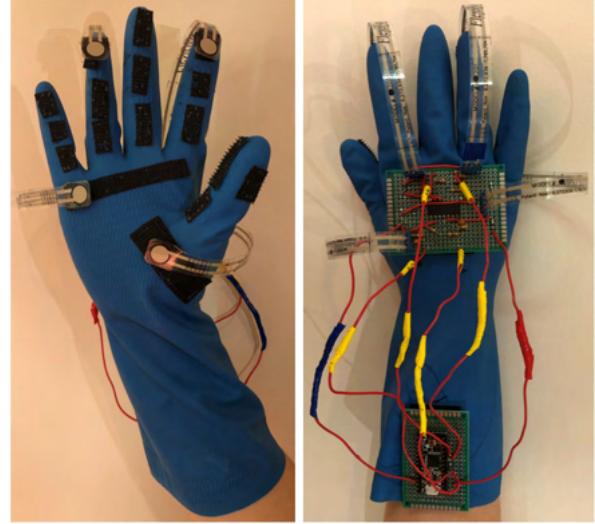


Fig. 1. Front and back of glove.

capture or force plates, however neither of these methods are accessible or adaptable as they require lab facilities and can be expensive. We aim to develop a glove that can measure forces experienced by the hands during dynamic motions and can be adapted to a variety of situations such as use on the human hand or robotic end effector configurations. Although force sensing gloves have previously been developed for commercial and research applications, most models are prohibitively expensive with prices of several thousand dollars¹ or do not reach the force ranges necessary to measure dynamic movement [2]. In this paper, we review how we developed our force sensing glove, the performance of the glove under various dynamic movement tests and strategies for improving the glove's future performance.

II. RELATED WORKS

A. Current Force Sensing Glove Models

Many gloves that have force sensing capabilities already exist, however most of these gloves are designed for use within the medical field and therefore are aimed at collecting measurements focused on hand positioning. One such device is the Neofect Smart Glove², which was developed for use in rehab and uses flex sensors to measure forces applied by the movement of the hand. While this may provide the user with

¹pressureprofile.com/body-pressure-mapping/tactile-glove

²www.neofect.com/us/smart-glove

detailed data on hand motion, it is not suited for measuring contact forces. Other devices that are currently available do not measure force ranges required for dynamic motion. For example the glove developed in Sundaram et al. [2] which can measure contact forces, but has a range of 30 mN to 0.5 N, which is not suitable for studying dynamic motions as they require a force range of above at least 10 N. Gloves such as the TactileGlove from Pressure Profile Systems³ can measure forces near the needed range, although still too low for many applications. However these gloves are not affordable, costing about \$25,000, making them inaccessible to most.

B. Flexible Sensing Technology

In developing this glove we required flexible sensors that could move with the hand during dynamic motions. One such flexible sensing technology measures force by sensing the deformation of magnetic particles placed on the area of interest [3], [4]. While this method does allow sensing across a large surface, it cannot measure forces at the needed ranges. Other flexible sensors use liquid metals which can act as inductors [5], which comes with the benefit of not needing rigid wiring in place and therefore has a high level of durability, however these methods can currently only detect whether there is force present, not the magnitude of the force being applied. Additionally, all sensors mentioned have only been developed on an experimental basis and are not available for commercial use, limiting their accessibility. We ultimately chose to use force sensitive resistors for our application as they are the most readily accessible from commercial retailers and can measure in the force ranges required for dynamic motion.

III. METHODS

A. Circuitry and Code

The force sensors used for the glove are the FlexiForce A201. The FlexiForce A201 is a force sensitive resistor with a standard force measurement range of 445 N. These sensors act as a variable resistor with an initially infinite resistance that decreases linearly as force is applied. The sensors are 191 mm long with a 9.53 mm diameter sensing area. In order to measure the force being applied to the sensor directly, we used a non-inverting op-amp circuit (Fig.2) that had been adapted from the FlexiForce Best Practices in Electrical Integration guide⁴. In this circuit, the sensor is connected to the negative voltage node of the MCP6004 op-amp IC with an 100 kΩ, 47 pF resistor-capacitor circuit connected across the negative voltage and output voltage nodes. Additionally, a voltage divider comprised of 4.7 kΩ and 1 kΩ resistors provides 0.58 V to the positive node of the op-amp. The op-amp uses the 3.3 V supply voltage and is grounded at V_{dd} .

We chose this circuit as it provides a better linear relationship between the resistance of the sensor and the applied

force than the simpler voltage divider or resistor-capacitor circuit and is less complex to implement than an inverting op-amp circuit which requires a dual voltage source. The op-amp circuit connects to an Adafruit ItsyBitsy M4 Express which is powered via USB. We chose to use the ItsyBitsy due to its size, as it is only 35.9 x 17.8 mm. This allows it to more comfortably fit on the hand, allowing movement to remain unimpeded.

Measurements were collected from the sensors using the Arduino IDE coding environment. Raw analog values were read from the ItsyBitsy inputs and converted into voltage readings (1), which were then converted into resistance values (2). This resistance value was then used to find the conductance of the sensor (3) which was then related to the applied force via the linear equation generated by the calibration process discussed below (4). This allowed for a live reading of applied forces on the sensors.

$$V_{out} = Analog_{out} * 3.3/1024 \quad (1)$$

$$R_s = R_{ref} * V_{ref} / V_{out} \quad (2)$$

$$G = 1/R_s \quad (3)$$

$$F = (G - b)/m \quad (4)$$

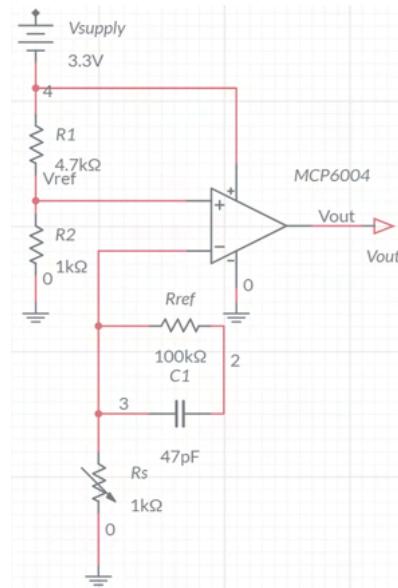


Fig. 2. Diagram of the non-inverted op-amp circuit used.

B. Glove Construction

The gloves are constructed out of dishwashing gloves, which were chosen due to their lightweight but durable material and thin, smooth surface which ensures the forces experienced by the hand are accurately represented by the sensor readings. The sensors are attached using Velcro to allow flexibility in their placement, thus allowing them to

³pressureprofile.com/body-pressure-mapping/tactile-glove

⁴www.tekscan.com/flexiforce-integration-guides

be optimized for different tests depending on the needs of the user. The Velcro is sewn on to the glove across each finger, across the upper palm and on the ball of the hand. We chose a standard configuration of four sensors attached to the ring and index fingers, ball of the hand and upper palm. These positions were chosen to allow measurement across the entirety of the hand structure and were found by performing ink tests (Fig.3) which provided a visual representation of the parts of the hand that were in most contact while the palm was open against a flat surface.

The circuitry of the glove is attached to the back of the hand so it does not interfere with natural movements during testing. The microcontrollers are housed on the wrist of the glove so as to ensure the USB wiring does not impede movement.



Fig. 3. Ink test used to determine optimal sensor placing.

C. Pucks

In order to ensure that all forces experienced by the sensors are evenly distributed across the sensing surface, we developed pucks which are cylinders with a diameter equal to that of the sensing surface (Fig.4). When applied to the sensors these pucks allow for the force to be directed evenly across the sensing surface and away from the edges of the sensor. This is because the sensor edges do not provide a linear relationship between the conductance of the sensor and the applied force. We tested several different types of pucks with varying degrees of compliance to determine how the compliance of the puck material affected the force sensing capabilities of the sensor. We tested pucks made from PLA, TPU-85 and felt furniture feet for these compliance tests.

In order to ensure that the differences in puck material was compensated for, we re-calibrated each sensor for the different puck materials (Fig. 5) using the calibration process described in section III-D.

D. Calibration

To allow the sensors to effectively convert resistance to force, it was required they be calibrated in the desired force ranges. This was achieved by collecting force data from the NexTech DFS500 force gauge using serial monitoring and comparing it with the measured conductance values from



Fig. 4. Pucks used in testing. From left to right, PLA, TPU and felt.

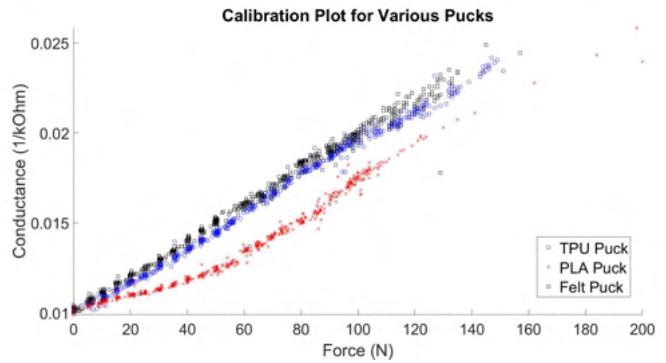


Fig. 5. Calibration plots of TPU, PLA and Felt pucks.

the force sensors. We related force to conductance values as for lower force ranges the sensors are more obviously linear when comparing conductance to force versus comparing resistance to force. Sensors were first conditioned by applying a force of 200 N and maintaining until the sensor output stabilized. The sensors were tested in a range of 0-140 N at intervals of 5 N every 5 seconds, however at forces above 110 N these intervals were more variable due to the limitations of the tester, therefore introducing more noise into those readings. These readings were then processed using MATLAB to plot conductance in units of $1/k\Omega$ and force in units of newtons, from which a linear trend line was generated (Fig.6). The equation of the trend line was then input into the Arduino code which allowed the applied force to be read directly from the sensor. The applied force readings to the sensor were not exact, generally reading 2-5 N off from the values displayed by the force gauge at forces below 110 N and 10-15 N off at forces above 110 N. This is likely due to the increased noise present in the calibration of higher forces and the inherent linearity error of the sensors, which was $\pm 3\%$.

IV. RESULTS

We tested three different kinds of dynamic motions using the felt pucks; leaning against a wall, pushing against a wall and grasping an object while in motion. The motion components of these tests are represented in figure 14. For all

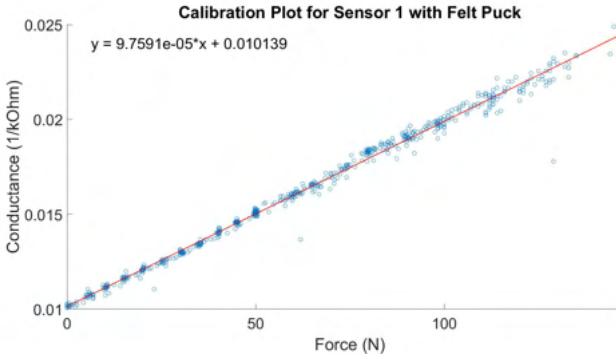


Fig. 6. Calibration plot for Sensor 3 using a felt puck. Conductance $1/k\Omega$ is on the y-axis and force in Newtons is on the x-axis.

three of these tests we found that the forces did not exceed about 25 N, and were generally evenly distributed across each sensor. We also tested variations in sensor placement and puck compliance. For all tests, the data was run through a low pass filter with a cutoff frequency of 2 Hz to allow general trends to be seen.

A. Wall Leaning

For the wall leaning test we observed higher forces present in the index finger, which reached 20 N of force as well as the upper palm and ball of the hand, which both reached about 15 N of force. The ring finger had a lower maximum force of 8 N. However it should be noted that the upper palm peaked at 15 N but was on average reading about 10 N, while the ball of the hand had a more steady 15 reading. Additionally the force on each sensor increased and decreased almost instantaneously and we observed that each sensor achieved a maximum force and remained generally constant at that value for the duration of the motion, creating an almost square waveform (Fig. 7).

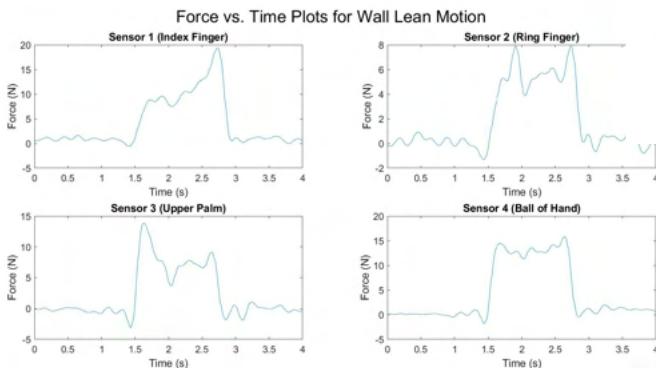


Fig. 7. Force vs. time plot for the wall lean motion.

B. Wall Push

For the wall push motion we recorded similar force ranges as in the leaning motion with a similar distribution of higher and lower forces (Fig. 8). Once again the forces on the sensors increased and decreased almost instantaneously,

however we recorded the sensors maintaining force measurements for a much shorter period of time than the leaning motion, representing the period of time that the hand was in contact with the wall for each test.

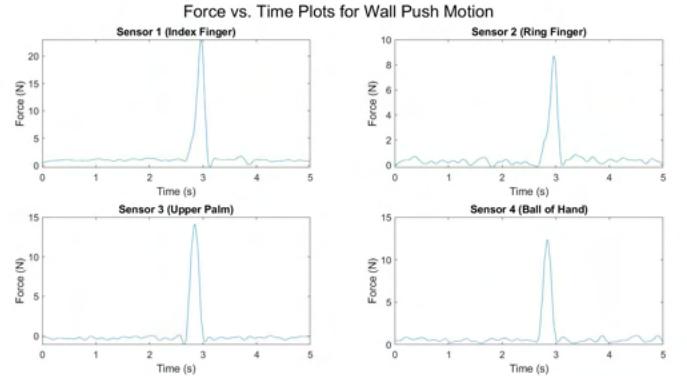


Fig. 8. Force vs. time plot for the wall push motion.

C. Dynamic Grasp

For the dynamic grasping motion we found that the force was generally concentrated on the fingers, with the upper palm not recording any force and the ball of the hand only recording a maximum of 4 N (Fig. 9). We also observed that while the increase in force is again near instantaneous, the decrease occurs over a period of about 3 seconds, which corresponds to the amount of time the object is grabbed for.

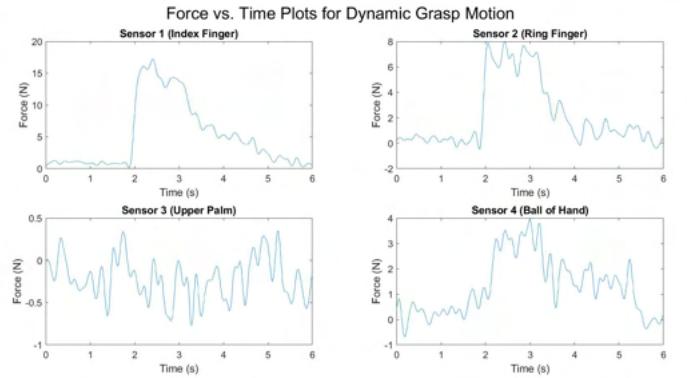


Fig. 9. Force vs. time plot for dynamic grasping motion.

D. Sensor Placement

We tested various sensor placements on the gloves in order to determine the effects on the overall force profile of the hand. We found that placing all the sensors on the palm of the hand generally decreased the amount of force experienced by the sensors with the peak force of the control trial being 20 N compared to around 12 N for the palm trial (Fig. 10). However, we did find that when the sensors were placed on only the fingers, the maximum force increased to around 23 N and all sensors recorded higher maximum force values than the palm only trial (Fig. 11). This indicates that more force is concentrated on the fingers during leaning motions. We also found that the data for these trials was

noisier than the data collected when the sensors were in the control arrangement.

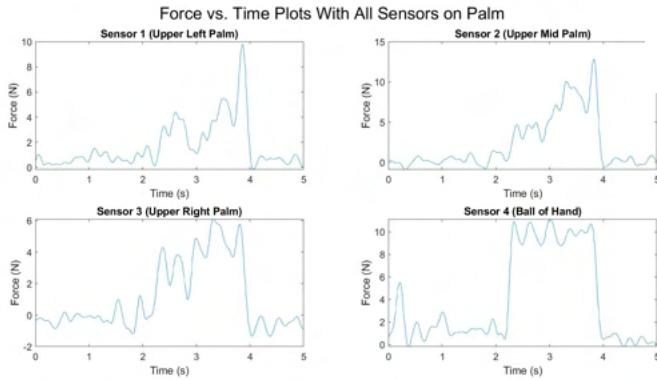


Fig. 10. Force vs. time plot for the leaning motion with all sensors affixed to the palm.

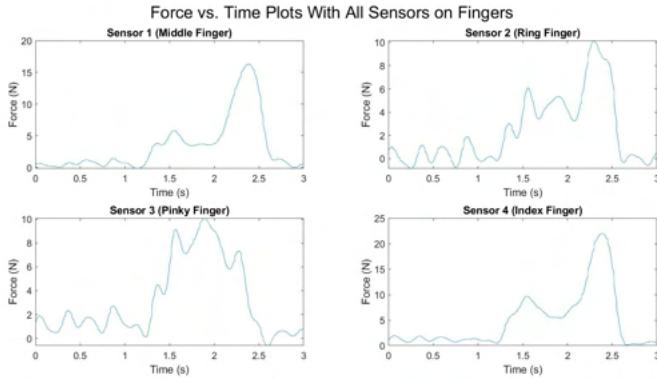


Fig. 11. Force vs. time plot for the leaning motion with all sensors affixed to the fingers.

E. Puck Types

We found that the compliance of the puck being used did not have a significant effect on the maximum force read by the sensors during the leaning motion, with all three pucks reading about 15 N of force (Fig. 12). However we did observe that the less compliant TPU and PLA pucks recorded a more noisy signal than the felt (Fig. 13). This may indicate that less compliant pucks are more sensitive to small changes in force than more compliant ones, as they absorb less force.

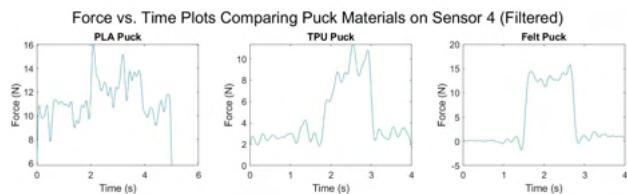


Fig. 12. Filtered force vs. time plots for PLA, TPU and felt pucks.

V. DISCUSSION

We found that our glove was satisfactorily able to measure the forces experienced by a human hand during a variety

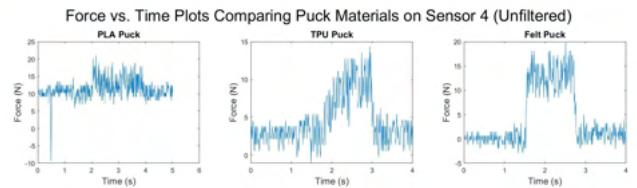


Fig. 13. Unfiltered force vs. time plots for PLA, TPU and felt pucks.

of dynamic motions. We were somewhat surprised at the force ranges observed during these tests, as previous testing had shown that some dynamic motions could generate forces of up to 100 N, however the low force ranges could be attributed to limitations in testing space which prevented extreme motions and differences in the forces measured by devices focusing on the entire hand versus individual fingers and parts of the palm. Additionally, the data collected from the glove had significant noise present. This could be due to the difficulties with calibration affecting the precision of the gloves and may be solved or diminished by improving this process. We were also surprised by the results of the puck tests, as the increase in signal noise between the puck materials was extreme. This could be due to several factors such as differences in puck calibrations or an increase in sensor sensitivity from less compliant materials. This presents an interesting focus for possible future study to determine why this occurred and further explore how puck material choice may impact the use of the glove.

VI. CONCLUSION

Through experimentation, we were able to determine that the force sensing glove we developed meets all of our requirements. Although it is not as precise in its force measurements as it could be, it provides a good overview of the forces experienced by the human hand during dynamic motions. Additionally, it has shown to have excellent adaptability as it is able to change the sensor configuration based on the needs of the user, allowing for it to be used in a greater range of testing situations. The sensors can also be calibrated for use with different hand and end effector types, for example if the sensor is affixed to a compliant material meaning that they can be used in a variety of testing environments. As the sensors are flexible, they could easily be integrated into a soft robotic system as well, further expanding the gloves possible uses. The use of easy to acquire, relatively cheap components in the development of this glove means that it is a more accessible option for measuring force than current comparable products, even if there is a possible trade off in the precision of the measurements. However, future work could include developing a more regulated calibration process by using a compression testing machine to better maintain force values for extended periods of time and continuing to test puck materials in the calibration and testing phases to determine optimum puck material use. As well, the circuitry could be adapted to accommodate more sensors to

allow for a more defined picture of the force distributions across the hand.

REFERENCES

- [1] T. Kwon, Y. Lee, and M. Van De Panne, "Fast and flexible multilegged locomotion using learned centroidal dynamics," *ACM Transactions on Graphics*, vol. 39, no. 4, July 2020. [Online]. Available: <https://dl.acm.org/doi/10.1145/3386569.3392432>
- [2] S. Sundaram, P. Kellnhofer, Y. Li, J.-Y. Zhu, A. Torralba, and W. Matusik, "Learning the signatures of the human grasp using a scalable tactile glove," *Nature*, vol. 569, no. 7758, pp. 698–702, May 2019. [Online]. Available: <http://www.nature.com/articles/s41586-019-1234-z>
- [3] T. Hellebrekers, O. Kroemer, and C. Majidi, "Soft Magnetic Skin for Continuous Deformation Sensing," *Advanced Intelligent Systems*, vol. 1, no. 4, p. 1900025, 2019, eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/aisy.201900025>. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/isy.201900025>
- [4] T. P. Tomo, A. Schmitz, W. K. Wong, H. Kristanto, S. Somlor, J. Hwang, L. Jamone, and S. Sugano, "Covering a Robot Fingertip With uSkin: A Soft Electronic Skin With Distributed 3-Axis Force Sensitive Elements for Robot Hands," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 124–131, Jan. 2018, conference Name: IEEE Robotics and Automation Letters.
- [5] S. Hamaguchi, T. Kawasetsu, T. Horii, H. Ishihara, R. Niiyama, K. Hosoda, and M. Asada, "Soft Inductive Tactile Sensor Using Flow-Channel Enclosing Liquid Metal," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4028–4034, July 2020, conference Name: IEEE Robotics and Automation Letters.

APPENDIX



Fig. 14. Photo sequences of dynamic motions. From top to bottom, leaning motion, wall push motion and dynamic grasp.

Bill Of Materials				
Part	Number	Unit Price	Price	
FlexiForce A201 Sensor (4 Pack)	x1	\$75.50	\$75.50	
Adafruit ItsyBitsy M4	x1	\$14.95	\$14.95	
MCP6004 Op-Amp Integrated Circuit	x1	\$0.53	\$0.53	
Various Resistors	x12	\$0.10	\$1.20	
47 pF Capacitor	x4	\$0.28	\$1.12	
Latex Free Dishwashing Glove	x1	\$2.38	\$2.38	
PCB Board	x2	\$0.34	\$0.68	
Amphenol FCI Clincher Connector (3 Position, Male)	x4	\$0.95	\$3.80	
Wiring	N/A	\$2.50	\$2.50	
Total Price:				\$102.66

Fig. 15. Bill of materials with total cost for the glove.

ACKNOWLEDGMENT

The authors would like to thank the Robotics Institute Summer Scholars program and Carnegie Mellon University as well as the National Science Foundation for providing the resources and funding to make this research possible. They would also like to thank RISS coordinators Rachel Burcin and Dr. John Dolan for their support.

Randomized Approach to Informative Path Planning for multiple UAVs

Rachel Moan¹, Brady Moon², and Sebastian Scherer³

Abstract—Unmanned Aerial Vehicles are rapidly becoming the go-to method for reconnaissance tasks such as search and rescue. In order to perform these missions, reconnaissance UAVs must be able to quickly locate people, vessels, or other targets given a limited flight time. This problem, known as informative path planning or orienteering, is essential to the success of search and rescue missions. Unfortunately, it is NP-hard and most present state-of-the-art solutions are generally too slow to present an effective solution online. Recent work in [1], however, presented RAOr: a solution to the Orienteering Problem that offers significant improvement over prior methods. This paper presents an analysis of RAOr and determines the types of environments and circumstances in which it may offer improvements over other accepted methods, such as a greedy algorithm. Additionally, we present a version of RAOr that works with multiple UAVs.

Index Terms—Search and Rescue Robots, Path Planning

I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) are rapidly becoming the go-to method for search and rescue operations. Due to their size and maneuverability, they are well-suited for navigating dangerous terrain quickly and safely, whereas it would be much slower and riskier for a human to complete the same task. In recent years, UAVs have been used to survey areas affected by Hurricane Katrina [cite] and Hurricane Wilma [2]. These UAVs, however, were operated by human, and were not capable of executing their missions autonomously. Autonomous UAVs would eliminate the need for constant human supervision, making the search and rescue process more efficient.

In order to perform these types of missions, reconnaissance UAVs must be able to quickly locate people, vessels, or other targets given limited battery or time constraints. This problem, known as Informative Path Planning (IPP) or the Orienteering Problem, is essential to the success of search and rescue missions. Unfortunately, it is NP-hard [3], and most present state-of-the-art solutions are generally too slow to present an effective solution online. As a result, human operators must control the vehicle using a video stream, which is not as effective as an autonomous UAV would be.

Recent work in [1] proposed Randomized Anytime Orienteering (RAOr): a solution to the Orienteering Problem that offers effective solutions quickly enough to be used online, rivaling the solutions found using other common algorithms like RIG, GCB, and a Greedy approach. This algorithm,

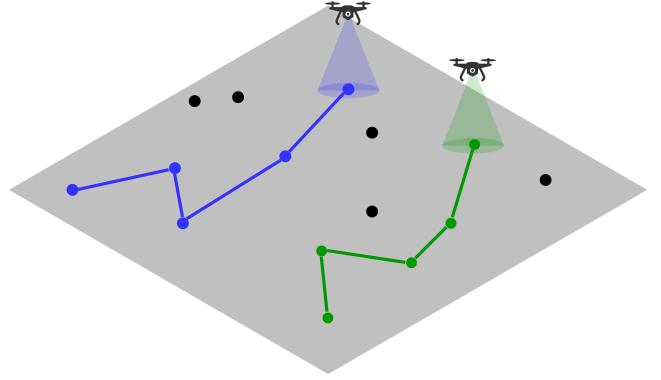


Fig. 1: Two UAVs searching nodes in an environment.

which has outperformed state-of-the-art methods in experimentation, could potentially allow for an autonomous reconnaissance UAV to be used for search and rescue missions.

The goal of this work is to expand upon RAOr by developing a version of it that can be used on multiple UAVs simultaneously. We also explore the various conditions and parameters that affect the performance of RAOr. In particular, we attempt to determine the different types of environments in which RAOr outperforms other algorithms, such as the Greedy approach.

The rest of this paper is organized as follows: we first describe the relevant literature in section II. We then move on to describe our methods in section III and our experiment results in section IV. We wrap up our work in section V by discussing our future work.

II. RELATED WORK

One of the most significant challenges faced by search and rescue UAVs is finding an effective way to navigate a given environment, locating as many targets as possible, while staying within a certain budget. In other words, an effective solution to the Informative Path Planning Problem (IPP) or Orienteering Problem is essential to the success of search and rescue missions.

Unfortunately, this problem is NP-hard [3], but this is a common problem that has been studied extensively by past researchers. [4] proposed a recursive greedy approach that finds approximate solutions. Another approach called the Generalized Cost Benefit (GCB) proposed in [5] maximizes a submodular function subject to a cost constraint.

Prior work in [1] established a new solution to the Orienteering problem called Randomized Anytime Orienteering

¹Rachel Moan is with the Department of Computer Science at Winthrop University. moanr@winthrop.edu

²Brady Moon is with the Robotics Institute at Carnegie Mellon University. bradym@andrew.cmu.edu

³Sebastian Scherer is with the Robotics Institute at Carnegie Mellon University. basti@andrew.cmu.edu

(RAOr) that appears to offer improvements upon the aforementioned approaches. RAOr works by sampling a set of nodes and generating a route. It then chooses a random and, if the node is not in the route, adds the node to the route. If the node is in the route, then RAOr removes it. Then the algorithm determines if the new route is better than the old one, and keeps it if it is. This appears to be a favorable solution for the Orienteering Problem for several reasons. Mainly, evidence suggests that it can outperform the current state of the art solutions.

What RAOr lacks, however, is an analysis of the conditions in which it would prove to be more useful instead of other methods, such as the greedy algorithm. This paper provides insight into these conditions. In particular, we examine how different environments affect the performance of RAOr. Additionally, RAOr does not currently have a version that supports multiple UAVs. This paper extends upon the original RAOr algorithm to apply it to multiple UAVs at once.

III. METHOD

A. Multi-RAOr

1) Algorithm Overview: Multi-RAOr takes as input the following:

- m : The number of routes that will be found for the environment.
- V : A list of the node in the current environment.
- $S = [s_1, s_2, \dots, s_m]$: the starting node for each path.
- $E = [[e_1^1, e_1^2, e_1^3, \dots], [e_2^1, e_2^2, e_2^3, \dots], \dots [e_m^1, e_m^2, e_m^3]]$: A list of the end nodes for each path.
- B : the maximum budget of each path.
- T_r : the maximum runtime of the algorithm.

The Multi-RAOr algorithm works by first partitioning the input nodes, V , into m sets. This creates a list of sets of nodes, N (line 1). Each of these sets represents the nodes that are particular UAV is responsible for covering. Then for each set of nodes, a random set of those nodes is selected that contains the specified start and end nodes for that UAV. The order in which these nodes should be visited is found by running the TSP solver (line 6). If the generated route is within the budget constraint, B , it is saved as the best route.

Then, for each set of nodes, $n \in N$, a new random node is selected (line 15). If this node is already in the current route for this set of nodes, it is removed from the route (lines 16-18). Otherwise, the node is added to the route (lines 19-21). Next, if the route is within the budget and its reward is higher than the reward of the current best route, this route becomes the best route, r_{best} .

This process is repeated until time runs out ($T_c > T_r$). Then the algorithm returns a list m routes.

2) Node Partitioning: The nodes are partitioned based on location. Alg. 2 describes how the nodes were divided. The partitioning algorithm takes a list of nodes in the environment, V , and divides them into m sets of nodes, where m is the number of UAVs available. They are partitioned in the order that they are input, meaning that the first

Algorithm 1: Multi-RAOr

Data: $m, V, S = [v_1, v_2, \dots, v_m], E = [v_1, v_2, \dots, v_m], B, T_r$

Result: A list, R , containing the best route for each UAV.

```

1  $N = Partition(V);$ 
2  $R = \emptyset;$ 
3 for  $i = 1 : length(N)$  do
4    $n = N(i);$ 
5    $s = SampleSet(n);$ 
6    $R(i) = TSP(n, v_s^i, v_e^i);$ 
7    $R(i)_{best} = \emptyset;$ 
8   if  $RouteLength(R(i)) \leq B \wedge Reward(R(i)) > Reward(R(i)_{best})$  then
9     |  $R(i)_{best} = .R(i);$ 
10   end
11 end
12 while  $T_c < T_r$  do
13   for  $i = 1 : length(N)$  do
14     |  $n = N(i);$ 
15     |  $v_{new} = Sample(n);$ 
16     | if  $IsInRoute(R(i), v_{new})$  then
17       |   |  $r = DeleteFromRoute(R(i), v_{new});$ 
18     | end
19     | else
20       |   |  $r = AddToRoute(R(i), v_{new});$ 
21     | end
22     | if  $RouteLength(r) \leq B \wedge Reward(r) > Reward(R(i)_{best})$  then
23       |   |  $R(i)_{best} = r;$ 
24     | end
25   end
26 end
27 return  $R$ 

```

$m/length(nodes)$ will be assigned to the first set, and so on. In our case, the nodes were generated in order of location, so each node was relatively close to the other nodes in its respective set.

B. Experimentation

In order to determine the factors that affect the performance of RAOr, we ran a series of experiments. We explored two primary factors that affected the performance of RAOr: the altitude of the UAV and the randomness of the node selection.

1) Budget: The most obvious factor that affects the reward of certain route is budget. Intuitively, we know that a higher budget yields better results. If a search and rescue UAV has a longer battery life, it can spend more time locating targets and collecting a higher reward.

In an ideal scenario, reconnaissance UAVs would have infinite battery life and time to complete their mission. Unfortunately, budgets are not infinite, and are often incredibly

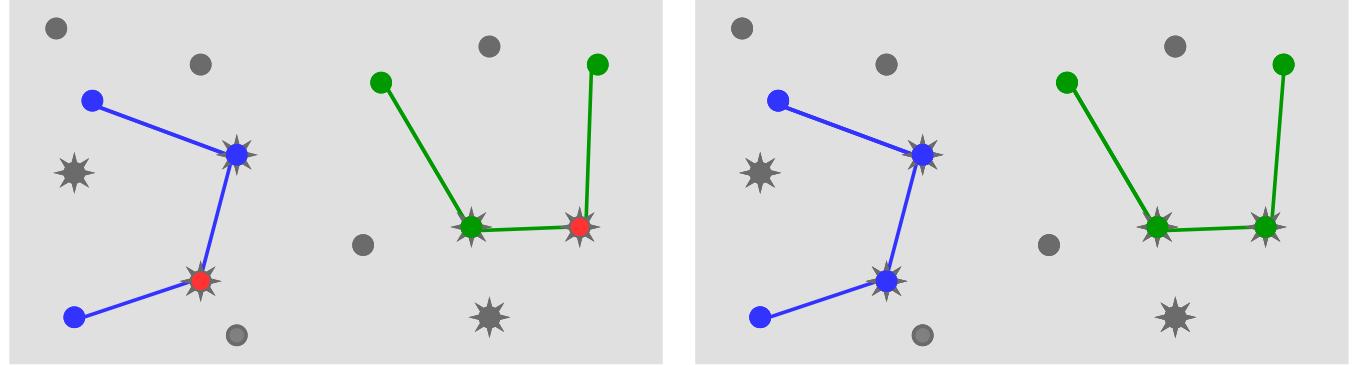


Fig. 2: An illustration of how multi-RAOr (Algorithm 1) would work with two UAVs. The paths of the UAVs are shown in blue and green, respectively. First, a set of nodes is sampled and a path is generated using a TSP solver (a). Then a random node is sampled for each path (b). Then the node is added to its respective path (c). If the new path is within budget and it has a higher reward than the current best path, it becomes the new best path (d).

Algorithm 2: Node Partitioning

Data: V, m
Result: N , a list of m sets of nodes.

```

1  $N = \emptyset;$ 
2  $index = 1;$ 
3  $num = ceil(length(V)/m);$ 
4 for  $i = 1 : length(V)$  do
5    $index = floor(i/num) + 1;$ 
6    $N(index) += V(i);$ 
7    $N(index) = Append(N(index), V(i))$ 
8 end
9 return  $R$ 
```

limited, especially when you have a larger environment to search.

2) *Altitude of nodes*: In addition to budget constraints, we also looked at the height of the sensors. Travelling to higher altitudes could give a reconnaissance UAV more information. It would be able to see more nodes, and therefore be able to identify more targets. However, if it gets too high, it might lose information.

IV. RESULTS

A. Multi-RAOr Demo

Figure 3 demonstrates the results of an example run of multi-RAOr. We ran multi-RAOr on environment with several low valued nodes clustered together, and three high-valued nodes farther away. The low-valued nodes are worth 0.1 and the high valued nodes are worth 10. In order for the reward to be acceptable, RAOr has to hit all three of the high-valued nodes. It was run with two agents, a budget of 70 meters, and a runtime of 20 seconds.

B. Features affecting RAOr's success

1) *Budget*: In order to determine the effect of the budget on the performance of RAOr, we ran several simulations of RAOr while varying the budget from 50 to 150. We also ran a greedy algorithm on the same environment and compared the results. Figure 4 shows the results of this experiment.

With a lower budget, RAOr outperforms the Greedy algorithm. However, when the budget increases past 120, Greedy starts to yield a higher reward. This shows that RAOr is well-suited for scenarios in which a UAV has a smaller budget, relative to the environment. If you need to search a large area but do not have ample time to do so, RAOr will generally

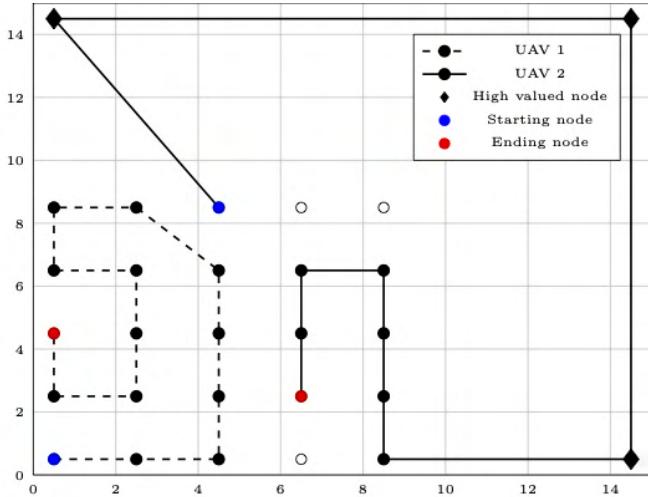


Fig. 3: This graph shows the result of an example run of multi-RAOr. Higher valued nodes are depicted with a diamond shape. Standard nodes are depicted by circles. An empty circle indicates that a node has not been covered by either UAV.

lead to a higher reward. However, if a UAV does have a large budget relative to the size of the environment, then the Greedy algorithm will result in a higher reward.

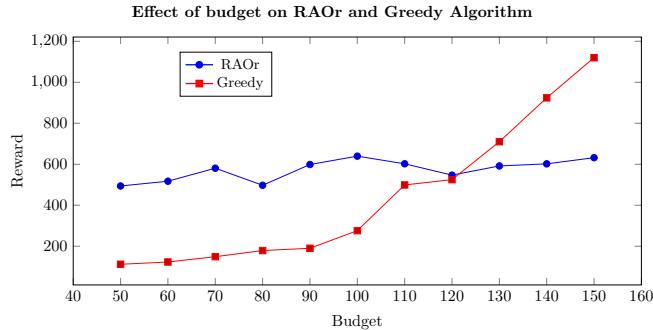


Fig. 4: This graph compares the effect of changing the budget (m) on the reward. We compared both RAOr and a Greedy Algorithm. This graph shows the average of 10 trials.

2) *Altitude of the nodes:* To examine the benefit of higher altitudes, we ran a simulation with two types of nodes: low sensors and high sensors. The height of the low sensors was held constant while the height of the high sensors was varied.

The results in figure 3 show that going to higher altitudes does offer improvements in the overall reward, the optimal height of the higher sensors lying around 70 units. The reward significantly drops after that, and the UAV begins to lose information as it goes too high.

V. CONCLUSION AND FUTURE WORK

RAOr is a solution to the Orienteering problem that could make autonomous multi-UAV missions faster and more robust. This paper laid the groundwork for the first version of multi-RAOr, but there are still several areas left

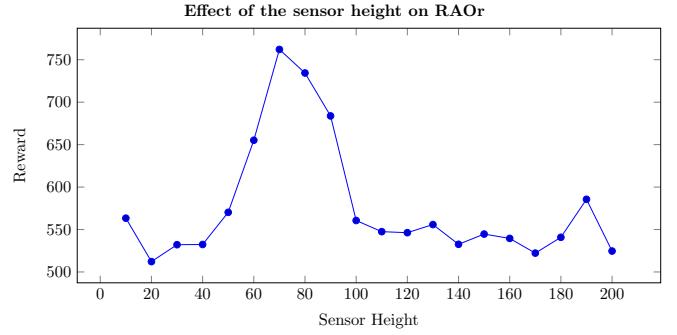


Fig. 5: This image shows the affect of higher sensors on the average reward of 30 trials.

to be explored. In particular, the way that the nodes are divided among routes could be improved. In our version, we partitioned the nodes based on their location. There are, however, other possible heuristics that could yield better results. For example, the nodes do not need to be split up at the beginning. Instead, we could allow every to access all the nodes, but only allow routes to select nodes that are not being used by another route.

The environment in which a UAV is deployed will significantly impact the performance of the algorithm used to solve the orienteering problem. This paper began to explore some of those factors. In the future, we plan to use the features of the environment such as sensor height and locations of highly-valued nodes to predict whether or not RAOr is an appropriate solution for the given context. These features could also be used to determine which algorithm would be best suited for different scenarios out of a set of options, similar to the ensemble planner described in [6].

ACKNOWLEDGMENT

This work was funded by the National Science Foundation grant no. 1659774. I would like to specially thank my mentors, Brady Moon and Dr. Sebastian Scherer, and the directors of the RISS program, Rachel Burcin and Dr. John Dolan, for their support throughout the summer.

REFERENCES

- [1] S. Arora and S. Scherer, "Randomized algorithm for informative path planning with budget constraints," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 4997–5004.
- [2] S. M. Adams and C. J. Friedland, "A survey of unmanned aerial vehicle (uav) usage for imagery collection in disaster research and management," in *9th international workshop on remote sensing for disaster response*, vol. 8, 2011, pp. 1–8.
- [3] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Research Logistics (NRL)*, vol. 34, no. 3, pp. 307–318, 1987.
- [4] C. Chekuri and M. Pal, "A recursive greedy algorithm for walks in directed graphs," in *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*. IEEE, 2005, pp. 245–253.
- [5] H. Zhang and Y. Vorobeychik, "Submodular optimization with routing constraints," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.
- [6] S. Choudhury, V. Dugar, S. Maeta, B. MacAllister, S. Arora, D. Althoff, and S. Scherer, "High performance and safe flight of full-scale helicopters from takeoff to landing with an ensemble of planners," *Journal of Field Robotics*, vol. 36, no. 8, pp. 1275–1332, 2019.

Facial Expressions in Quori using ROS and Gazebo

Allison Moore¹, Roshni Kaushik², and Dr. Reid Simmons²

¹Tufts University

²Robotics Institute, Carnegie Mellon University

Abstract— Peer tutoring can be as beneficial for the learning process of both teacher and student. However, while past work has looked into the role of robots as teachers, there is little research on using robots on the receiving end of tutoring. To address this gap, our work develops robots equipped to act as students and facilitate communication with a human teacher. Nonverbal facial expressions and gestural cues play an important role in traditional teacher student interactions. In this paper, we present a framework for communicating customizable facial features and expressions through ROS (Robot Operating System) for the humanoid robot, Quori. Ideally, these facial expressions will foster engagement between the robot and user, and improve communication through contextual information. In the long run, they will likely be one modality of many used to connect and engage with the user. This framework will depend on an array of customizable facial features that can be positioned to create unique faces. From there, these features will be set on motion paths to allow users to create a range of expressions through ROS commands.

I. INTRODUCTION

Facial Expressions play a defining role in human interaction and the communication of emotional states. In education, they are a significant contributor to the success of interactions between teacher and student. Past work has identified the use of facial expressions in conveying information helpful in collaboration such as emotion, motivations, and attitudes [12].

In our research, we generate a set of dynamic facial expressions on the humanoid robot Quori[13].

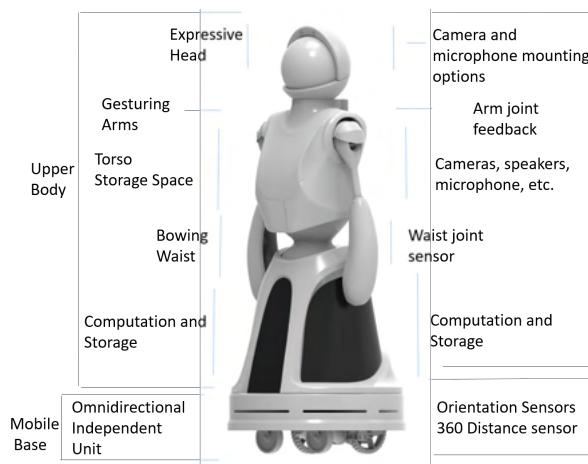


Fig. 1. Diagram of Quori and its components

Individually, this contributes to improving the state-of-the-art in human-robot interaction by introducing a new mode of communication to the Quori robot platform.

On a broader scale, this contributes to a larger ongoing project of developing a robotic student for use in peer tutoring environments. While there is a strong background of research on robots acting in a teaching capacity, there is a gap in the development of robots who can act as students. Despite the lack of current research in this field, teachable robots can serve as helpful tools within a classroom as peer tutors. Past work indicates that peer tutoring is linked to solidifying learning and deeper understanding [12], however in many environments it is difficult to find appropriately skilled and convincing pupils. Robotic students can fill this gap in the classroom to create more fulfilling learning experiences from peer tutoring. Given the importance of facial expressions in an educational context developing a range of customizable and interpretable facial expressions for Quori provides support for developing a robotic peer tutoring system.

Quori used a retro-projected head that can display a wide range of 2-dimensional facial expressions. To capitalize on this, we have developed a system of individually controllable facial aspects to create a variety of custom faces through ROS positioning of the features. Similarly, we reference Ekman and Friesen's Facial Action Coding System (FACS) [4] mapping these features to generate expressions understandable by a broad audience.

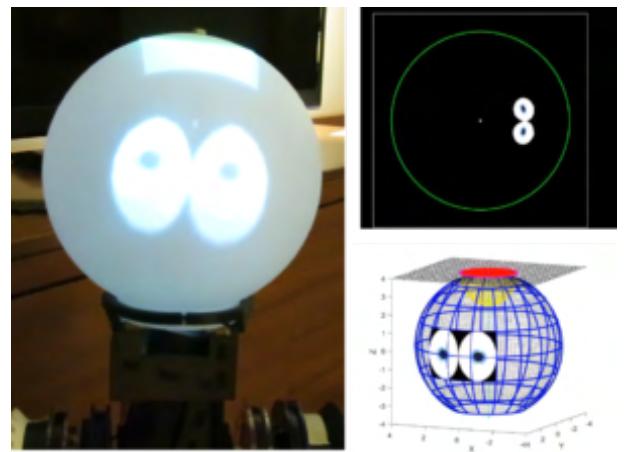


Fig. 2. Retro-projection system used to display facial expressions on Quori

II. RELATED WORK

Facial expressions are used broadly in social robotics. However, there is less research specifically on the impact of robot expressiveness specifically in an educational context. To build a set of comprehensible facial expressions, we referenced the Facial Action Coding System (FACS) developed by Ekman and colleagues.

A. Facial Action Coding System (FACS)

In FACS, facial expressions are divided into quantized action units (AUs) linked to facial movements [4]. Ekman used a coding system to identify quantitative measurements for universally acknowledged facial expressions. There is past work applying FACS to design robotics faces [15]. However, since Quori's face is a projected 2D image rather than a mechatronic setup, our use of the system is likely more similar to past research implementing FACS. In this area, there is significant work on computer synthesis models and dynamic animation[1], [14]. However, few examples integrate a FACS model into a robotic system, instead of having the generated model act as an independent agent.

B. Dynamic and Static Facial Expressions

Past studies on emotion recognition indicate that people begin to recognize facial expressions before the expression is fully formed [7]. Similarly, higher levels of brain activity occur in people responding to dynamic facial expressions rather than static images [10]. This increased depth of recognition led to our pursuit of a dynamic system where Quori can readily shift between expressions through a dynamically generated animation.

C. Emotional Expression in Classrooms

While there is more work on the correlation of teacher emotions to student engagement, it is likely a relationship also exists for the converse. Hagen Auer et. al. identified varying patterns in teacher emotions relative to their students' engagement where positive emotions in teachers seemed likely to induce positive emotions in students. With this in mind, it is plausible to assume that the perceived engagement of a robot student will impact the mood of a peer tutor and their willingness to teach [6]. Similar work has identified a high correlation between a supportive presentation style in teaching and positive affect in students whereas excessive demands on students link to negative affect[5]. In the context of the Quori robotic student, these patterns may be useful in mapping realistic reactions for Quori in future work .

III. METHODOLOGY

We developed a system of individual controllable facial features that project into a full face for the Quori robot. In a typical workflow, users input a desired facial expression by calling a JSON file with position and image path data for the moments associated with that expression. This file is processed by a base positioning function that opens and places features in OpenCV to create a series of composite face images sent to the robot as an "imgmsg." After the robot

runs the expression, the user can choose either to exit the program or continue with a new facial expression.

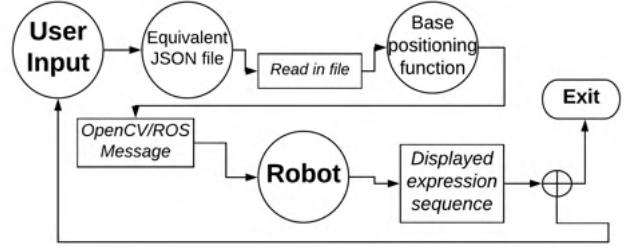


Fig. 3. Control flow path for feature commands to the robot

A. Facial Feature Development

In the current iteration of the facial features for Quori, there are ten distinct features: left eyebrow, right eyebrow, left eyelid, right eyelid, left pupil, right pupil, left light spot, right light spot, upper lip, and lower lip. The pupils, light spots, and nose can be resized, moved, and tilted but have one static image shape. The eyelids, mouth, and eyebrows have animated image sequences in addition to the ability to move and resize to give the appearance of motion. The shape and appearance of the features for this program are kept relatively simple to avoid an "uncanny valley" [8] appearance and make Quori seem friendly and approachable to student teachers. These features are based on those linked to emotional expression by Ekmen et. al [4]. With this in mind, the initial facial features for Quori are relatively simple with a cartoon like appearance. Past work indicates that cartoon features activate the same recognition pathways as human faces at slightly different timing intervals [11] and possibly a faster initial processing speed [16]. Since cartoon faces seem to produce reliable emotional interpretation without appearing unsettling in the way a realistic face would, we pursued a relatively simple look for our facial design.

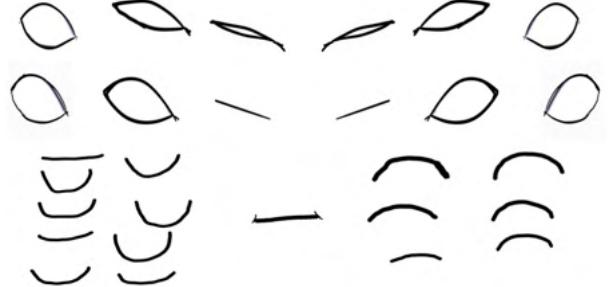


Fig. 4. Sample of some of the facial components (top: eyelids, bottom:lips) used to create composite facial expressions for Quori

B. Face Assembly

Quori's face is composed of distinct dynamically controllable parts that can combine into a single face animation

sequence. At present, the motion sequences and positions for each facial feature can be stored in a JSON file. We use OpenCV to place each sub-image for each frame into a larger composite facial image. From this instance, the sequence of generated faces is translated into a ROS parsable format and projected onto Quori's face. While we choose to use one of a cohesive library of simple facial features for our work, any image or sequence of images could be positioned and projected to Quori's face using this method.

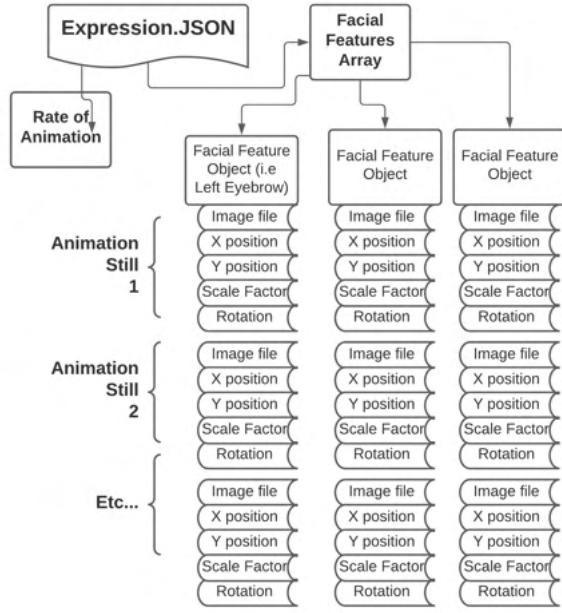


Fig. 5. System diagram of file structure for JSON specification of an animation sequence for Quori

Each JSON file is separated into several key/value pair objects. As depicted in Fig. 5, at the simplest level, each file contains an animation rate under the key name "transitionTime" and an array of "feature" objects under the key name "Features".

```
{
    "transitionTime":0.3,
    "Features": [
        {
            "leftEyeStillsPupil": [
                {
                    "file ":"images/pupil.png",
                    "x_multiplier":0.34,
                    "y_multiplier":0.33,
                    "scale_factor":0.5,
                    "rotation_factor":0
                }
            ]
        },
        {
            "Lower_Lip": [
                {
                    "file ":"mouth1.png",

```

```

                    "x_multiplier":0.5,
                    "y_multiplier":0.5,
                    "scale_factor":0.6,
                    "rotation_factor":0
                }
            ],
            "file ":"mouth2.png",
            "x_multiplier":0.5,
            "y_multiplier":0.5,
            "scale_factor":0.6,
            "rotation_factor":0
        }
    ],
    Continued ...
}
```

The "Features" array can contain the subelements: "leftEyeStillsPupil", "leftEyeStillsSpot", "leftEyeStillLid", "Upper_Lip", "Lower_Lip", "rightEyeStillsPupil", "rightEyeStillsSpot", "rightEyeStillsLid", "leftBrow", and "rightBrow". However, not all facial elements have to be included. Each element of the "Features" array contains a subarray of objects specifying the position of a specific facial feature locations for each frame. If only one frame is specified it repeats through the animation.

Within each frame, five parameters can be controlled: The image shown ("file"), the x position ("x_multiplier"), the y position ("y_multiplier"), the scale factor ("scale_factor"), and the rotation factor ("rotation_factor").

IV. RESULTS

With our current framework for broadcasting expressions to Quori, we have developed a beginner library of dynamic facial expressions to demonstrate Quori's ability to display basic facial cues. A starter set of facial features for Quori using this system appears in Fig. 7. The depicted images are screencaps from a fluid system of feature movement for each dynamic expression. Each of the expressions depicted in Fig. 7. cycles from a neutral to a full emotive expression. In each expression, we individually position the eyes, mouth, brows, and eyelids to mimic the muscle movements identified by Paul Ekman in his 1997 work on FACS [4].

The current list of test behaviors developed for this study is in Fig. 6. We reference each of these base expressions from the six basic emotions identified by Ekman [3] as well as base face states (blinking). We describe the associated facial movements for each expression in the second column of the table. We applied these descriptions to design the appearance of each of the expressions in Fig. 7.

A. Control System Development

In addition, to create these expressions we have developed two more generalized tools for sending face expression commands to Quori: a library of individual facial features (Fig. 4.) and a program for specifying movement paths, location, and timing for the allocation of said features. These two tools provide a system for development of additional facial behaviors on an operator side through controlling parameters

Expression	Associated Facial Movements[2]
Joy	Lip Pulled Up, Mouth Open
Contempt	Lip Corner Pulled Down, Brow Lowered, Lids Tightened
Surprise	Outer Brow Raised, Inner Brow Lowered, Lids Tightened, Lips Stretched
Distress	Lip Corner Depressed, Brow Lowered, Inner Brow Raised
Anger	Lids Tightened, Lips Tightened, Brow Lowered, Upper Lid Raised
Blink	Lids Tightened

Fig. 6. Current Emotional Cues programmed for Quori facial expressions and their associated movement paths

in a JSON file. Ideally, this should make programming facial behaviors for Quori accessible since specifying facial animation paths with this method can be done with little or no prior knowledge of coding.

While Quori still does not have the full range of facial expressions available to a human, this system lets Quori mimic standard expressions [9] and display a similar range of motion to that of a human face. In addition, this system of processing facial commands likely allows for more detail and feature complexity than used in the current demo.

V. CONCLUSIONS

In this work, we generate a system for creating distinct and dynamic facial expressions for the Quori robot. This relies on Python, ROS, and OpenCV to create a library of facial feature movements and expressions. We reference the Facial Acting Coding System (FACS) to help make these more comprehensible to users. By developing customizable facial expressions for Quori, we hope to increase Quori's effectiveness and likeability when communicating with users. Future work here may include a study validating the impact of Quori's facial expressions in this capacity. Our current assumption that the use of facial expressions will improve Quori's ability to communicate is based primarily on prior research regarding dynamic facial expressions and FACS[4]. However, since Quori is a unique platform and we are developing a unique set of facial expressions, their effectiveness merits further investigation. Similarly, the current command system for Quori's facial expressions requires explicit specification for each moved feature of every frame within the animation. In future work, it would be useful to investigate auto-generation between images to make the program set up less cumbersome. Moving away from the use of pre-drawn features would also provide more flexibility when specifying commands to Quori.

VI. FUTURE WORK

We hope to continue this work both by improving the current facial control system and by tying the current system into the larger-scale project of developing teachable robots.

A. Improvements on current system

Presently, the still images of Quori's features (i.e. eyelids, lips) are hand-drawn and must be individually specified between frames of motion by users. As this system develops we would like to move toward a more autogenerated approach where the frames between start and end positions can be drawn programmatically instead of simply assembled programmatically.

On a similar note, the current system of specifying the location of facial features for Quori makes it difficult to determine where realistic positions are and where the bounds of movement for a given facial feature should be. Since the positions for facial features are specified by a numeric multiplier, it can be difficult to intuitively know where this multiplier rests in relation to the face. We would like to be able to pre-specify these bounds of realistic motion for users to make the process of designing motion paths within the face easier.

Since Quori's facial features are unique to each use case and must be compiled individually for each facial expression sequence, loading an expression sequence can take several seconds. In anticipation of using this program for real-time interaction in the future, we would also like to investigate ways to speed up the compilation process or save facial expressions for Quori in a memory-effective manner.

B. Use in teachable robots

This work is part of a long-term project seeking to develop teachable robots that can act like students in a peer tutoring context. Within this framework, facial cues are one of several modes of expression that Quori is likely to use. We hope to pair the expressions and facial control software outlined in this work with body language and natural language processing cues to allow Quori to communicate with depth and clarity. In addition, we hope to develop Quori's expressions suit a peer tutoring environment better and to adapt Quori's facial expressions to match the verbal, gestural, or facial communication cues of participants interacting with Quori.

VII. ACKNOWLEDGEMENT

This work is supported by National Science Foundation Grant #1659774. I would also like to sincerely thank Roshni Kaushik, Dr. Reid Simmons, and Peter Vanegas for their guidance and support throughout this program, as well as the rest of the RASL Lab. Additional thanks to Rachel Burcin and John Dolan for their excellent management of the RISS Program.

REFERENCES

- [1] M. H. Alkawaz, D. Mohamad, A. H. Basori, and T. Saba. Blend shape interpolation and facs for realistic avatar. *3D Research*, 6(1):6, 2015.

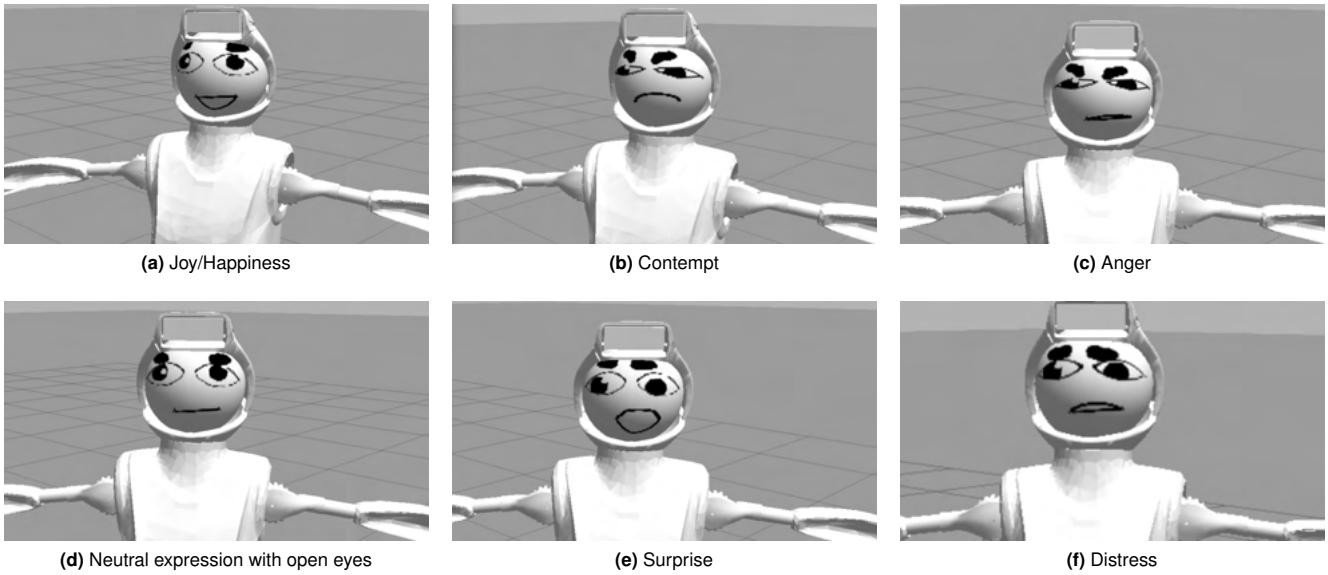


Fig. 7. Basic facial expressions demonstrated by Quori

- [2] J. F. Cohn, Z. Ambadar, and P. Ekman. Observer-based measurement of facial expression with the facial action coding system. *The handbook of emotion elicitation and assessment*, 1(3):203–221, 2007.
- [3] P. Ekman. Basic emotions. *Handbook of cognition and emotion*, 98(45-60):16, 1999.
- [4] P. Ekman and E. L. Rosenberg. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA, 1997.
- [5] T. Goetz, O. Lüdtke, U. E. Nett, M. M. Keller, and A. A. Lipnevich. Characteristics of teaching and students' emotions in the classroom: Investigating differences across domains. *Contemporary educational psychology*, 38(4):383–394, 2013.
- [6] G. Hagenauer, T. Hascher, and S. E. Volet. Teacher emotions in the classroom: associations with students' engagement, classroom discipline and the interpersonal teacher-student relationship. *European journal of psychology of education*, 30(4):385–403, 2015.
- [7] G. Horstmann and U. Ansorge. Visual search for facial expressions of emotions: A comparison of dynamic and static faces. *Emotion*,

- 9(1):29, 2009.
- [8] S. Y. Kim, B. H. Schmitt, and N. M. Thalmann. Eliza in the uncanny valley: Anthropomorphizing consumer robots increases their perceived warmth but decreases liking. *Marketing letters*, 30(1):1–12, 2019.
 - [9] D. Mazzei, N. Lazzeri, D. Hanson, and D. De Rossi. Hefes: An hybrid engine for facial expressions synthesis to control human-like androids and avatars. In *2012 4th IEEE RAS & EMBS International Conference on biomedical robotics and biomechatronics (BioRob)*, pages 195–200. IEEE, 2012.
 - [10] E. D. Ross, C. I. Prodan, and M. Monnot. Human facial expressions are organized functionally across the upper-lower facial axis. *The Neuroscientist*, 13(5):433–446, 2007.
 - [11] S. Schindler, E. Zell, M. Botsch, and J. Kissler. Differential effects of face-realism and emotion on event-related brain potentials and their implications for the uncanny valley theory. *Scientific reports*, 7(1):1–13, 2017.
 - [12] M. Shayganfar, C. Rich, and C. L. Sidner. A design methodology for expressing emotion on robot faces. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4577–4583. IEEE, 2012.
 - [13] A. Specian, N. Eckenstein, M. Yim, R. Mead, B. McDorman, S. Kim, and M. Matarić. Preliminary system and hardware design for quori, a low-cost, modular, socially interactive robot. In *Workshop on social robots in the wild*, 2018.
 - [14] K. R. Thorisson. Toonface: A system for creating and animating interactive cartoon faces. *Learning and Common Sense Section Technical Report*, pages 96–01, 1996.
 - [15] T. Wu, N. J. Butko, P. Ruvulo, M. S. Bartlett, and J. R. Movellan. Learning to make facial expressions. In *2009 IEEE 8th International Conference on Development and Learning*, pages 1–6. IEEE, 2009.
 - [16] J. Zhao, Q. Meng, L. An, and Y. Wang. An event-related potential comparison of facial expression processing between cartoon and real faces. *PloS one*, 14(1):e0198868, 2019.

Learning a CBF controller for Adaptive Cruise Control Under Model Uncertainty

Emanuel Munoz¹, Qin Lin², and John M. Dolan²

Abstract—The Adaptive Cruise Control (ACC) problem is a common scenario for autonomous vehicles where safety is a critical consideration. Through solving this issue, traffic flow could be enhanced while assuring the safety of passengers. Promising solutions based on Control Barrier Functions (CBF) have been previously proposed to assure safe performance. These methods formulate safe invariant sets that must be well-defined based on the dynamic model. In real scenarios, a high-precision model is not available because of unknown uncertainties produced by uncontrollable parameters like rolling resistance. A CBF based on an uncertain ACC model could provoke a collision because its formulation depends on the model. A recent work was able to diminish the model uncertainty based on an episodic learning approach with successful results on similar systems. Inspired by this learning method, we propose a learning CBF controller that can reduce the effects of the uncertain parameters in an ACC system. The solution includes using a neural network estimator updated with aggregated data. We validate the controller performance for uncertain rolling resistance in a vehicle simulation.

Index Terms—adaptive cruise control, safety, uncertainty

I. INTRODUCTION

Autonomous vehicles are an innovative promising solution to traffic flow, safety, pollution, etc. in populous cities [1]. Vehicles require advanced hardware and software integration to perform autonomous driving while assuring safety. In particular, a car in a road scenario faces several situations that endanger the passenger's safety. The Adaptive Cruise Control (ACC) problem is a common situation for vehicles where safe behavior is critical [2]. The problem essentially consists of maintaining a safe distance from a front vehicle while trying to reach an arbitrary desired velocity. ACC represents an important issue in highways where dangerous collisions could occur if this problem is not adequately solved [3].

Implemented safe systems for the AAC problem in commercial vehicles have been discussed in previous works [4], [5]. A recent approach for stability and safe maneuver stands out in the robust control area. Control Barrier Functions (CBFs) [6] formulate a optimization problem where a safe performance is assured. CBFs have shown to be a promising solution for general systems [7]. The ACC problem was also treated under this method in previous works. In [8], the authors introduced a variant CBF method to solve the ACC in different scenarios. Also, [9] proposed a unification of the CBF method and Control Lyapunov Function (CLF)

¹ The author is with Electrical Engineering Department, Universidad de Ingeniería y Tecnología (UTEC), Lima, Peru

² The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, 15213 PA, USA

for assuring safety and stabilizing a cruise control system. CLF is an analogous method to CBF but is oriented to the stability of the system.

CBF formulation is based mostly on the dynamic model of the system. However, generally only uncertain models are available, so safe behavior can not be guaranteed [10]. For the ACC problem where safety is critical, additional approaches must be taken to diminish the effects of uncertainty. In particular, the uncertainty sources come from the rolling resistance, air drag, or inaccurate sensors.

Learning approaches are popular in the literature for approximate unknown or uncertain models. This type of method is based on collected data to estimate models or functions. In robust control, learning approaches have shown promising results when applied to assure safety. [11] proposes a safe reinforcement learning framework based on CBFs to perform safe control. Also in [12], the authors synthesize a learned CBF based on expert demonstration data. Similarly, [13] presents a synthesized CBF based on trajectory data. Particularly for uncertain models, some CBF works have discussed this problem. [14] presents a hybrid synthesized control to correct an uncertain system. In [15], a probabilistic approach is implemented to synthesize a CBF in control affine systems based on Gaussian Processes. [16] proposes a novel reinforcement learning framework based on CBF and CLF for uncertain systems applied to biped robots.

In a recent work [17] dealing with model uncertainty, the authors provide a straightforward implementation using episodic data aggregation. This work uses the Data Aggregation method (DAgger) [18] to collect adequate data. Given that it is not feasible to obtain data from the whole input-state space, DAgger fundamentally collects data from experimental trials so the estimator can be improved each time.

In this work, we propose an episodic-learned controller based on CBF and CLF to safely drive a system while stabilizing to the desired path. We implement this solution specifically for the Adaptive Cruise Control problem, where we assume the rolling resistance parameters are uncertain. We are inspired by the work in [17] in that we collect data episodically to use it for estimation of the model. Our contribution is to safely control the ACC system with uncertain parameters. We compare the performance of our improved controller and a nominal model-based controller to show the improvements.

The remainder of this paper is structured as follows. First, we introduce the necessary notations for CBFs and CLFs in Sec. II. Second, we state the ACC problem notation and

safety and stability conditions while also referring to the model in Sec. III. Third, we present the episodic formulation and the implemented algorithm in Sec. IV. Fourth, we present a comparison and analysis of the proposed method in Sec. V. Finally, we present some conclusions and further discussion for future work in Sec. VI.

II. CONTROL LYAPUNOV FUNCTIONS AND CONTROL BARRIER FUNCTIONS

In this section, we introduce CLF and CBF notations. CLF and CBF have similar formulations, but each is for a different purpose. We do not intend to give a full review, but instead only the information needed for the current work. Also, we present the formulation of the Quadratic Programming (QP) problem formulated from both functions (CBF and CLF), so a feasible solution can be found. The notation provided is based on the previous work [6].

A. Control Lyapunov Functions

Lyapunov functions emerged as a tool for stability analysis for nonlinear systems. Consider in particular a time-invariant affine system as follows

$$\dot{x} = f(x) + g(x)u \quad (1)$$

where $x \in \mathcal{X} \subset \mathbb{R}^n$ is the n -dimensional state system, $u \in \mathcal{U} \subset \mathbb{R}^m$ is the m -dimensional control input of the system. Let us also define $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ and $g : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times m}$ as continuous Lipschitz functions. We assume the control task is to stabilize the system such that $x(t) \rightarrow x_e$. We define a continuous and differentiable function V called Lyapunov such that $V : \mathcal{X} \rightarrow \mathbb{R}^+$ is bounded and $V(x_e) = 0$. Particularly, we focus on an *exponentially Lyapunov function* such that the system stabilizes asymptotically to x_e as

$$\exists u = k(x) \text{ s.t. } \dot{V}(x, k(x)) \leq -\kappa_V(V(x)) \quad (2)$$

where κ_V is a class \mathcal{K} function. In particular, we choose $\kappa_V(a) = \lambda a$ for a constant $\lambda > 0$. Also, the derivative \dot{V} is expressed as follows

$$\dot{V}(x, k(x)) = L_f V(x) + L_g V(x)k(x) \quad (3)$$

where $L_f V(x)$ and $L_g V(x)$ represent the Lie derivative of the system denoted as $\nabla V(x)f(x)$ and $\nabla V(x)g(x)$, respectively. We say that there is a *Control Lyapunov Function* if $\exists \lambda > 0$ such that

$$\inf_{u \in \mathcal{U}} (L_f V(x) + L_g V(x)k(x)) \leq -\lambda(V(x)) \quad (4)$$

has a solution. Therefore, the system is exponentially stable with a decay rate λ for all $x \in \mathcal{X}$.

B. Control Barrier Functions

CBFs are defined analogously to CLF, but for safety instead of stability. We define a continuous and differentiable

function $h(x) : \mathcal{X} \rightarrow \mathbb{R}$ such that its *superlevel set* $\mathcal{C} \in \mathbb{R}^n$ can be named as a safe set. Let the set \mathcal{C} obey

$$\mathcal{C} = \{x \in \mathcal{X} : h(x) \geq 0\} \quad (5)$$

$$\partial\mathcal{C} = \{x \in \mathcal{X} : h(x) = 0\} \quad (6)$$

$$\text{Int}(\mathcal{C}) = \{x \in \mathcal{X} : h(x) > 0\}. \quad (7)$$

We consider a system like (1) such that

$$\exists u = k(x) \text{ s.t. } \dot{h}(x, k(x)) \geq -\kappa_h(h(x)) \quad (8)$$

where $\kappa_h \in \mathcal{K}$ that is particularly chosen as $\kappa_h(a) = \gamma a$ for a constant $\gamma > 0$. Also, the time derivative of h is expressed as

$$\dot{h}(x, k(x)) = L_f h(x) + L_g h(x)k(x). \quad (9)$$

We say the function h is a CBF if there exists a $\gamma > 0$ for which

$$\inf_{u \in \mathcal{U}} (L_f h(x) + L_g h(x)k(x)) \geq -\gamma(h(x)) \quad (10)$$

for all $x \in \mathcal{X}$. The solution u assures that the set \mathcal{C} is invariant, therefore $x(t \rightarrow \infty) \in \mathcal{C}$.

C. QP formulation

CLF and CBF can be formulated as a quadratic program problem for a particular minimization problem for u . Assuming we have a nominal control signal u_{ref} , we can guarantee stability and safety through the following optimization problem

$$\begin{aligned} k(x) = \operatorname{argmin}_{u \in \mathcal{U}} & (u - u_{ref})^T H(u - u_{ref}) + p\delta^2 \\ \text{s.t. } & L_f V(x) + L_g V(x)u + \lambda(V(x)) \leq \delta \\ & \text{s.t. } L_f h(x) + L_g h(x)u + \gamma(h(x)) \geq 0 \end{aligned} \quad (11)$$

where σ is a slack variable that relaxes the system under the penalization $p > 0$ and $H \in \mathbb{R}^m \times \mathbb{R}^m$ is an arbitrary positive definite matrix. The addition of the slack variable is to relax the stability condition so the safety condition is also feasible.

III. ADAPTIVE CRUISE CONTROL PROBLEM

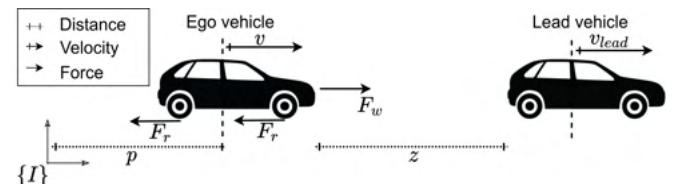


Fig. 1: Adaptive cruise control representation. Ego vehicle behind the lead vehicle in a one-dimensional motion.

In this section, we introduce the ACC problem. Consider two vehicles: a lead vehicle and an ego vehicle, as shown in Fig. 1. The lead vehicle is in front of the ego vehicle, while both are on the road. Assume the lead vehicle is traveling at a constant velocity v_{lead} . The control objective is to drive

the ego vehicle so it can reach a desired velocity v_d while maintaining a safe distance from the lead vehicle. The safe distance is not a constant because it usually depends on the velocity v of the ego vehicle. The ego vehicle has a position p measured from an inertial frame I . We also define the relative position of the lead vehicle through the distance z measured from the ego vehicle in a local frame L . Given the variables mentioned we define the state space and control space as

$$x = [p, v, z]^T \in \mathbb{R}^3 \quad u \in \mathbb{R} \quad (12)$$

where u is the input variable which represents the wheel force F_w of the ego vehicle. From basic kinematics and dynamics, we know that

$$\frac{d}{dt}z = v - v_{lead}, \quad m\frac{dv}{dt} = F_w - F_r$$

where m is the mass of the vehicle and F_r is the rolling resistance of the wheels. F_r can be modeled as a polynomial function

$$F_r(v) = f_0 + f_1v + f_2v^2 \quad (13)$$

where f_0 , f_1 , and f_2 are coefficients obtained empirically. Given this relationship we can formulate the affine state system similarly to [9] as

$$\dot{x} = \begin{bmatrix} v \\ -\frac{1}{m}F_r(v) \\ v_{lead} - v \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m} \\ 0 \end{bmatrix}u = f(x) + g(x)u. \quad (14)$$

Notice that the functions f and g have the same properties as (1). Also, both functions depend on parameters obtained empirically, which could introduce uncertainty into the model. In this work, we focus on the scenario where the coefficients in (13) are not known. Let us also consider that the input is limited by physical constraints of acceleration and deceleration. The control signal can be constrained linearly as $mc_{dg} \geq u \geq mc_{ag}$, where g is gravitational acceleration and c_a and c_d are acceleration and deceleration factors. Considering the control objective, we formulate a valid CLF and CBF for this affine system.

A. CLF

We define the Lyapunov function for the system (14) as

$$V(x) = (v - v_d)^2 \quad (15)$$

such that it is a positive definite function. Also, this function drives the system to the stability objective $v \rightarrow v_d$. The gradient of the function is obtained as

$$\nabla V(x) = [0 \quad 2(v - v_d) \quad 0]$$

so the Lie derivatives of (15) are calculated as

$$L_f V(x) = \nabla V(x)f(x) = -\frac{2}{m}F_r(v)(v - v_d)$$

$$L_g V(x) = \nabla V(x)g(x) = \frac{2}{m}(v - v_d)$$

Notice how each derivative still depends on the parameters of the system. Therefore, after replacing each function to have the same condition shown in (11), we finally have the following inequality

$$\begin{aligned} \dot{V}(x, u) + \lambda V(x) &= \\ (v - v_d) \left(\frac{2}{m}(u - F_r(v) + \lambda(v - v_d)) \right) &\leq \delta. \end{aligned}$$

B. CBF

The safety objective of the system is to maintain a safe distance z between the vehicles. First, let us consider that there is a lookahead time T_h (also called headway) such that $T_h v$ is less than the distance. Also, we know that there is a maximum braking distance for the vehicle to decelerate from $v \rightarrow v_{lead}$. Given these constraints, we define the barrier function h as

$$h(x) = z - T_h(v) - \frac{1}{2} \frac{(v - v_{lead})^2}{c_{dg}} \quad (16)$$

where the second term refers to the braking distance weighted by the minimum input possible. The gradient of h is expressed as

$$\nabla h(x) = [0 \quad -T_h - \frac{v - v_{lead}}{c_{dg}} \quad 1]$$

so that its Lie derivatives are

$$\begin{aligned} L_f h(x) &= \nabla h(x)f(x) = \frac{1}{m}F_r(v)(T_h + \frac{v - v_{lead}}{c_{dg}}) + (v_{lead} - v) \\ L_g h(x) &= \nabla h(x)g(x) = -\frac{1}{m}(T_h + \frac{v - v_{lead}}{c_{dg}}) \end{aligned}$$

Both equations when substituted into (11) give the following inequality

$$\begin{aligned} \frac{1}{m} \left(T_h + \frac{v - v_{lead}}{c_{dg}} \right) (F_r(v) - u) + (v_{lead} - v) + \\ \gamma \left(z - T_h(v) - \frac{1}{2} \frac{v - v_{lead}}{c_{dg}} \right) \geq 0. \end{aligned} \quad (17)$$

IV. EPISODIC UNCERTAINTY LEARNING

In this section, we present the proposed solution for the model uncertainty in the ACC system. First, we present the new formulation so an uncertain model can be used for the controller. We provide a general form for this case. Then, an algorithm for implementation is shown for the ACC system.

A. Formulation

Recalling (1), we assumed we know a perfectly defined system. However, a realistic implementation should consider that the functions f and g are only estimated, so the actual system used for the control formulation is

$$\hat{x} = \hat{f}(x) + \hat{g}(x)u \quad (18)$$

where \hat{f} and \hat{g} are the estimated functions. We can introduce the real functions f and g to this formulation with a simple arrangement as

$$\dot{x} = \hat{f}(x) + \hat{g}(x)u + (f(x) - \hat{f}(x)) + (g(x) - \hat{g}(x))u.$$

Notice that the functions are grouped such that we can replace them with error functions defined as

$$\epsilon_f(x) = f(x) - \hat{f}(x) \quad \epsilon_g(x) = g(x) - \hat{g}(x).$$

Therefore, for a calculation similar to (9), the derivative of the barrier function with respect to the state is expressed as

$$\dot{h}(x, u) = \frac{\partial h}{\partial x} \left(\hat{f}(x) + \hat{g}(x)u + \epsilon_f(x) + \epsilon_g(x)u \right).$$

Considering that we continue with the development shown in (9) with the estimated functions, the output of this formulation would not be the actual \dot{h} but instead a \hat{h} that is obtained from the estimated system (18) using (9). We can conveniently group the expression as follows

$$\dot{h}(x, u) = \hat{h}(x, u) + e_f(x) + e_g(x)^T u, \quad (19)$$

where we also replace the last terms such that

$$e_f(x) = \frac{\partial h}{\partial x} \epsilon_f(x) \quad e_g(x) = \frac{\partial h}{\partial x} \epsilon_g(x). \quad (20)$$

e_f and e_g are unknown functions in the expression. However, they can be approximated using a supervision learning method as long as there are sufficient data available. We assume we have a tuple dataset D defined as

$$D = \{(x_i, u_i, \hat{h}_i), \dot{h}_i\}_{i=1}^N,$$

containing N tuples composed of the state x and the input u with its respective \dot{h} and \hat{h} obtained. We can now define two parametric functions \hat{e}_f and \hat{e}_g that depend on the parameters $\theta \in \Theta$. Replacing the functions in (19), the expression is

$$\hat{S}(x, u, \theta) = \hat{h}(x, u) + \hat{e}_f(x, \theta) + \hat{e}_g(x, \theta)^T u$$

where $\hat{S}(x, u, \theta)$ is an approximation of \dot{h} . This new expression allows us to define a minimization problem such that there exists an optimal parameter θ^* that can be found from

$$\theta^* = \underset{\theta \in \Theta}{\operatorname{argmin}} \mathcal{L}(\hat{S}(x, u, \theta), \dot{h}(x, u)).$$

\mathcal{L} refers to an error function or loss function that measures the difference between $\hat{S}(x, u, \theta)$ and $\dot{h}(x, u)$. We can approximate θ^* using a supervised learning approach such as a neural network. Therefore, the quadratic programming formulation in (11) is replaced as follows

$$\begin{aligned} k(x) &= \underset{u \in \mathcal{U}}{\operatorname{argmin}} (u - u_{ref})^T H(u - u_{ref}) + p\delta^2 \\ \text{s.t. } L_f V(x) + L_g V(x)u + \lambda(V(x)) &\leq \delta \\ \text{s.t. } \hat{S}(x, u, \theta) + \gamma(h(x)) &\geq 0. \end{aligned} \quad (21)$$

B. Algorithm

For training, we require a large amount of data for the dataset D . As it was discussed on [17], it is possible to obtain data episodically. A known method in imitation learning called DAgger (Dataset Aggregation) proposes essentially to obtain data from the execution of a process using the estimated controller (policy), to label the data obtained, and then to aggregate the new data so an improved estimated controller can be updated [18]. This “closed-loop” data collection is useful if the control objective does not change significantly, such that it is not necessary to explore all the dataset D space. We propose Algorithm 1, which includes this idea to train the estimator \hat{e}_f and \hat{e}_g . Notice that the algorithm shown is inspired by the work [17], which presents a similar approach more generally.

Algorithm 1: Learning CBF controller

Input: CLF V , CBF h , initial estimators $\hat{e}_{f,i=0}$, $\hat{e}_{g,i=0}$, initial state set \mathcal{X}_0 , number of experiments T , time step dT , final time episode t_f

Output: Optimized estimators \hat{e}_f , \hat{e}_g

```

1  $D = \{\emptyset\}$ 
2 for  $i = 0$  to  $T$  do
3    $x_0 \sim \text{Uniform}(\mathcal{X}_0)$ 
4    $x = x_0$ 
5    $D_i = \{\emptyset\}$ 
6   for  $j = 0$  to  $t_f/dT$  do
7      $u_{ref} = F_r(v)$ 
8      $k, h, \hat{h} = \text{qp-controller}(x, u_{ref}, \hat{e}_f, \hat{e}_g)$ 
9      $x' = \text{dynamics}(x, k)$ 
10     $\dot{h} = (h_i - h_{i-1})/dT$ 
11     $D_i = D_i \cup \{(x, k, \hat{h}), \dot{h}\}$ 
12     $x = x'$ 
13   $D = D \cup D_i$ 
14   $J = \frac{1}{t_f T / dT} \sum \mathcal{L}(\hat{e}_f(D[x]) + \hat{e}_g(D[x])D[k] + \hat{h}, \dot{h})$ 
15   $\theta = \theta - \alpha \frac{dJ}{d\theta}$ 

```

We feed this algorithm a number of experiments T or episodes for collecting data, a time step dT since we implement the controller in discrete time, and a final time t_f for the end of the experiment. Also, we provide zero-initialized estimators $\hat{e}_{f,i=0}$ and $\hat{e}_{g,i=0}$ such that $\theta = 0$. The functions V and h represent the formulation presented in (15), (16). We should also set up a state space \mathcal{X}_0 from which we sample an initial state x_0 . Because this is a deterministic system, we set different initial states so more data could be collected.

The algorithm starts with an empty dataset D to be filled through experiences. We set T experiments, where we initialize x_0 and a temporary dataset D_i that collects data in an experiment. We run the experiment t_f/T times. We choose as an input reference the rolling resistance, so the vehicle can compensate for it. The algorithm also indicates a function `qp-controller()` which refers to the solution

Parameters	Values
dT	0.02 (s)
t_f	10 (s)
\mathcal{X}_0	[⟨0, 4⟩, ⟨18, 22⟩, ⟨38, 42⟩]
v_{lead}	22 (m/s)
v_d	24 (m/s)
m	1650 kg
$[f_0, f_1, f_2]$	[0.1, 5, 0.25]
c_a, c_d	0.3
T_h	1.8 (s)
η	2×10^{-2}
λ	5.0
γ	5.0
T	9
$\mathcal{L}()$	Mean Squared Error
Gradient descent optimizer	ADAM [19]
α	10^{-4}

TABLE I: Parameters used for simulation implementation and controller

of (21) and the calculations given from (16) and (19). Similarly, the function `dynamics()` refers to the real ACC system modeled in (14). It is possible to obtain the real \dot{h} from simple numerical differentiation of (16)'s result. The obtained data are appended to the temporary dataset, so it can be aggregated to D . We consider modeling the estimators \hat{e}_f and \hat{e}_g using a neural network so the rest of the procedure corresponds to the gradient descent algorithm. Notice that the algorithm defines a learning rate α for the gradient descent.

V. EXPERIMENTAL RESULTS

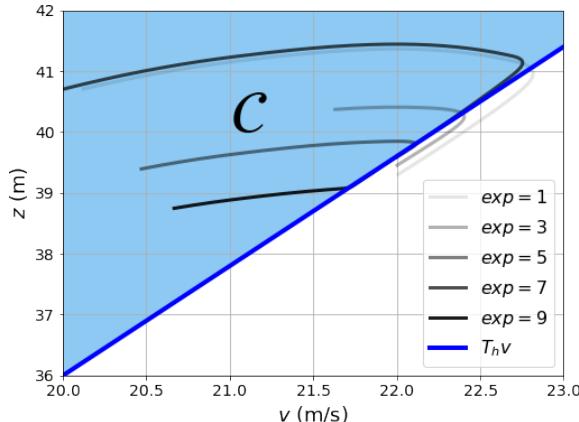


Fig. 2: Simulation results for odd-numbered experiments. Evolution of the states v and z . Blue line indicates barrier function. Blue shaded zone is the safe set \mathcal{C} .

We implemented Algorithm 1 in a configured numerical simulation based on (14). The parameters used in the simulation are shown in Table I. The estimators were modeled as feedforward neural networks of input size the same as the size of the state ($n = 3$) and two hidden layers of 50 nodes chosen arbitrarily. Each layer was configured to have a sigmoid activation function except for the output layer that returns the linear result directly. We used Torch as a base library for fast implementation. Additional configurations are also shown in Table I. Recalling the ACC real model is

different from the one used for the controller formulation, it is required to simulate uncertainty. Hence, for validation purposes we consider different nominal parameters f_0^* , f_1^* , and f_2^* for the rolling resistance. The values set in the simulation are $[f_0^*, f_1^*, f_2^*] = 10[f_0, f_1, f_2]$.

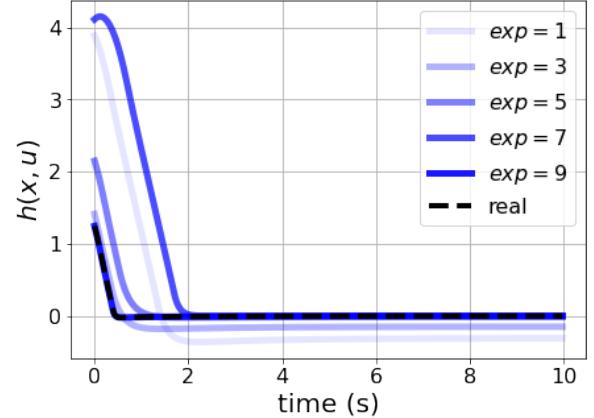


Fig. 3: Simulation results for odd-numbered experiments. Evolution of the control barrier function h through the experiments. Dashed black line represents the performance of h if the controller were perfectly known.

Fig. 2 shows the evolution of the states v and z through the experiments denoted as exp . For $exp = 1$ when no estimation was made, it is evident that the controller drives the system out of the safe set. We can validate the improvement of the system for the $exp = 9$ where we can assume the estimator had enough information. It is worth mentioning that trajectories for each experiment are different because the initial state is obtained from sampling. Notice that as more data are aggregated through the experiments, better performance of the controller is obtained.

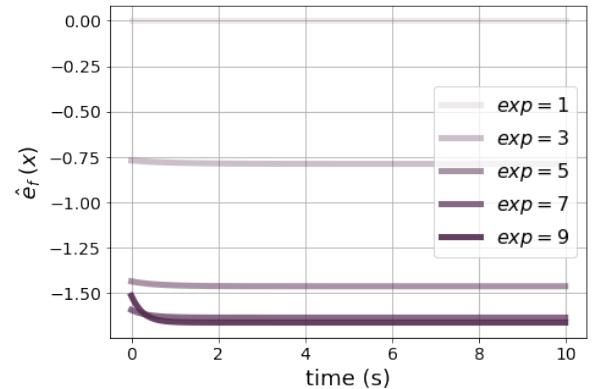


Fig. 4: \hat{e}_f results for odd-numbered experiments.

Similarly, Fig. 3 confirms the overstepping behavior of the safe set, such that it does not accomplish (6). The $h(x, u)$ curve in $exp = 9$ overlaps the form of the real system. We could say that the algorithm compensates for the error of the uncertainty.

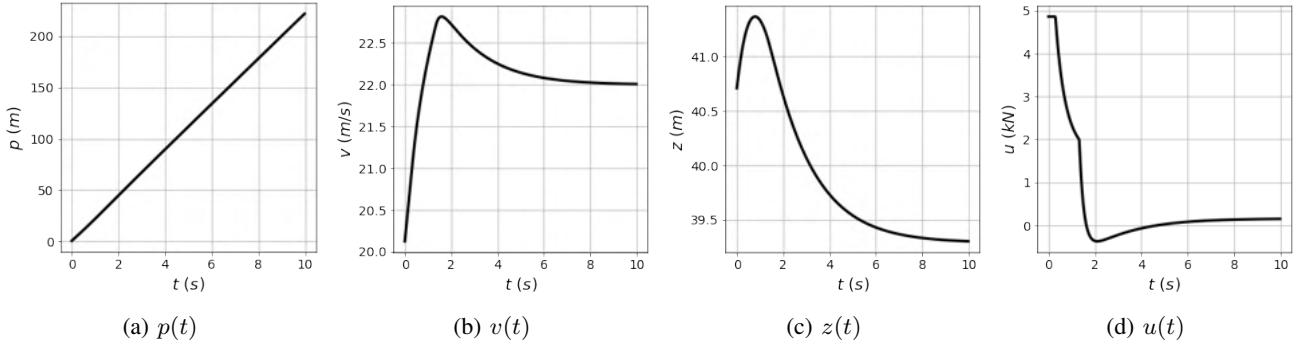


Fig. 5: Response in time of state and input signal for the experiment $exp = 10$

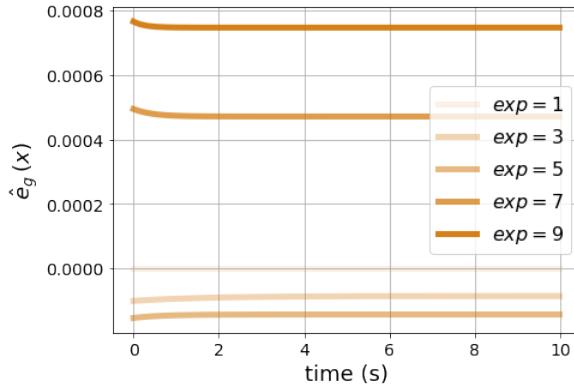


Fig. 6: \hat{e}_g results for odd-numbered experiments.

In Fig. 4, we show the evolution of \hat{e}_f in different experiments. Notice that because of zero initialization, the value for $exp = 1$ was zero. \hat{e}_f converges to an almost constant function through the experiments. This curve differs from the evolution of \hat{e}_g in Fig. 6, where the estimator converges to zero. Remember that the only parameters changed belong to the f function, so this behavior is expected. Also, the \hat{e}_f curves have a constant behavior that compensates for the uncertainty, but remember that F_r is a dynamical function that depends on v . In Fig. 4, there is a slight transition at the start of the experiment. The constant behavior of the curves can be justified because there are unbalanced data. Most of the data appear in a stable state, so the data could be filled mostly with constant values. However, even under this observation, it has been proved that the algorithm corrects the uncertainty, as is shown in Fig. 3. A better understanding of the estimator's behavior could be analyzed in more complex systems with more dynamical behavior.

We show in Fig. 5 the time response of the state and input for $exp = 10$. Notice in Fig. 5b that the trajectory tries at first to reach the desired velocity $v_d = 24$ m/s, but because of maintaining a safe distance, it decelerates. The plot in Fig. 5d confirms this behavior such that it reduces the velocity and stabilizes the safe distance shown in 5c. It is worth mentioning that the velocity variance does not change the linearity of the position p .

VI. CONCLUSIONS

We validate the performance of a learning-enabled CBF controller for an ACC application. Results show that the proposed controller reduces the error produced by parameter uncertainty in the model. The performance of the distance of the ego vehicle when using the controller is proved to assure safety better than using only a traditional CBF controller relying on a possibly inaccurate dynamic model. The estimators in the controller converged such that it adequately compensates the model uncertainty.

However, an imbalance in the dataset raises the question of whether the proposed controller will be adequate for more complex systems. Future work should also consider that an uncertain model could have more sources of error than only model parameters. The implemented estimators are deterministic functions that could not work well for Gaussian sensor noise. Stochastic estimators should also be tried under the same framework presented in this work.

ACKNOWLEDGMENT

This work was made with the support of the CMU Robotics Institute and the Robotics Summer Scholars Program. Emanuel would like to thank Qin Lin, John Dolan, Rachel Burcin, and the RISS community for their support during this research project.

REFERENCES

- [1] F. Duarte and C. Ratti, "The impact of autonomous vehicles on cities: A review," *Journal of Urban Technology*, vol. 25, no. 4, pp. 3–18, 2018.
- [2] G. Marsden, M. McDonald, and M. Brackstone, "Towards an understanding of adaptive cruise control," *Transportation Research Part C: Emerging Technologies*, vol. 9, no. 1, pp. 33–51, 2001.
- [3] A. K. Yadav and J. Szpytko, "Safety problems in vehicles with adaptive cruise control system," *Journal of KONBiN*, vol. 42, no. 1, p. 389, 2017.
- [4] E. Baschab, S. Ball, A. Vazzana, and J. Sprinkle, "Safer adaptive cruise control for traffic wave dampening," in *Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems*, 2021, pp. 231–232.
- [5] J. Lunze, "Adaptive cruise control with guaranteed collision avoidance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 5, pp. 1897–1907, 2018.
- [6] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3420–3431.

- [7] Z. Li, “Comparison between safety methods control barrier function vs. reachability analysis,” *arXiv preprint arXiv:2106.13176*, 2021.
- [8] W. Xiao, C. Belta, and C. G. Cassandras, “Adaptive control barrier functions for safety-critical systems,” *arXiv preprint arXiv:2002.04577*, 2020.
- [9] A. D. Ames, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs with application to adaptive cruise control,” in *53rd IEEE Conference on Decision and Control*. IEEE, 2014, pp. 6271–6278.
- [10] A. J. Taylor and A. D. Ames, “Adaptive safety with control barrier functions,” in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 1399–1405.
- [11] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, “End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 3387–3395.
- [12] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, “Learning control barrier functions from expert demonstrations,” in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 3717–3724.
- [13] N. M. Boffi, S. Tu, N. Matni, J.-J. E. Slotine, and V. Sindhwani, “Learning stability certificates from data,” *arXiv preprint arXiv:2008.05952*, 2020.
- [14] A. Robey, L. Lindemann, S. Tu, and N. Matni, “Learning robust hybrid control barrier functions for uncertain systems,” *arXiv preprint arXiv:2101.06492*, 2021.
- [15] P. Jagtap, G. J. Pappas, and M. Zamani, “Control barrier functions for unknown nonlinear systems using gaussian processes,” in *2020 59th IEEE Conference on Decision and Control (CDC)*. IEEE, 2020, pp. 3699–3704.
- [16] J. Choi, F. Castaneda, C. J. Tomlin, and K. Sreenath, “Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions,” *arXiv preprint arXiv:2004.07584*, 2020.
- [17] A. Taylor, A. Singletary, Y. Yue, and A. Ames, “Learning for safety-critical control with control barrier functions,” in *Learning for Dynamics and Control*. PMLR, 2020, pp. 708–717.
- [18] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.
- [19] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

Informative and Fast Exploration Planning Using UAV for Reconnaissance Operations

Kaleb Ben Naveed^{*1}, Brady Moon² and Sebastian Scherer²

Abstract—In the missions related to search and rescue operations, reconnaissance Unmanned Arial Vehicles (UAV) are used to effectively search the given environment map and return information about the detected objects with limited flight time. This involves solving the NP-hard problem of maintaining balance between the tasks of fast exploration and data acquisition. Most of the existing work focuses on optimizing only one of these factors. In this paper, we propose Prioritized-FUEL, which is built on top of the FUEL (Fast UAV Exploration) algorithm, a frontier-based exploration technique. The proposed hierarchical structure maintains balance between fast coverage and data acquisition through the introduction of two high-level planner options: Exploration planning and Informative planning. In order to facilitate decision making for informative planner, we modify Frontier Information Structure (FIS) in the original FUEL paper to incorporate information about objects of interest. Moreover, we introduce Frontier Priority Que (FPQ) to store information about all the frontiers, which have a higher probability of the presence of the objects of interest near them. The results from the experiments in the light UAV simulation environment show that the proposed method resulted in almost 2 times faster data acquisition as compared to the original FUEL algorithm.

Index Terms—Search and Coverage, Informative Path Planning, Fast Exploration

I. INTRODUCTION

Unmanned aerial vehicles (UAVs) are widely used for the purpose of data acquisition tasks. Some of the examples of these tasks include victim search in rescue operations [1], fault inspection for infrastructure [2], and water body exploration for ecosystem management [3]. The main dilemma for UAVs in these operations is maintaining a balance between the tasks of fast exploration and information gathering. This is specifically important in a scenario of search and rescue, where the UAV does not only has to explore the region faster but also has to periodically pause exploration to focus on potential interesting areas where a victim could be present.

Some of the work [4][5] proposed for the task of optimal and rapid exploration have shown great results in the real world settings but do not consider information gain. On the other hand, some of the work [6][7] proposed on informative path planning or uncertainty reduction, which have shown promising results by maximizing information gain during exploration tasks, lack fast coverage guarantees. Thus most of the existing work either focuses on rapid exploration or

¹Student of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hong Kong, China.
kaleb-ben.naveed@connect.polyu.hk

²The Robotics Institute, Carnegie Mellon University

* Supported by CMU Robotics Institute Summer Scholars Program

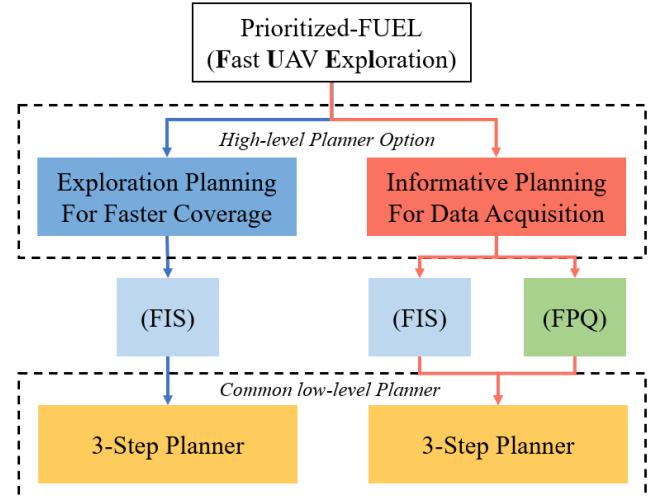


Fig. 1. The Prioritized-FUEL (Fast UAV Exploration) proposes a hierarchical structure with two high-level options: Exploration planning for faster coverage and Informative planning for data acquisition. High-level option of Informative planning uses **Frontier Priority Que (FPQ)** in addition to **Frontier Information Structure (FIS)** for decision making. At lower level, 3-step planner is used for trajectory generation. Both high-level options use same low-level 3-step planner. The only difference is the input to 3-step planner.

information gain, which limits their ability to provide optimal solutions in reconnaissance operations.

In order to solve this problem, we propose, Prioritized-FUEL, which is inspired from FUEL (Fast UAV Exploration) algorithm. The proposed method add the ability in existent FUEL framework to balance exploration and exploitation for information gain by using hierarchical structure, which contains two high-level planning options: Exploration planning for faster coverage and informative planning for data acquisition. The High-level option of exploration planning is responsible for the task of fast coverage of the search space, while the high-level option of informative planning is responsible for the task of information gathering of the detected objects. The selection of the high-level option is made based on the contents of Frontier Information Structure (FIS) and the proposed Frontier Priority Que (FPQ). FIS contains essential information about the search space and frontiers. We modify the original structure of FIS introduced in the FUEL algorithm [8] according to our decision making requirements. Details about modification can be found in Section III and Section IV.

The High-level option of Exploration planning only uses

FIS; however, the Informative planning option uses both FIS and FPQ for decision making and planning stage. The overall structure can be seen in Figure 1. The proposed FPQ stores information about the frontiers, which have an overlap with the bounding box of the detected objects. After the high-level planner option is selected, a 3-step planner is used for generating minimum-time trajectories. The three steps include creating optimal global paths by posing the problem as the Travelling Salesman Problem (TSP), refining local viewpoints for maximum coverage, and generating minimum-time B-spline trajectories which are safe, obstacle-aware, and dynamically feasible.

The main contributions of this paper can be summarised as follows:

- The hierarchical structure, which ensures balance between fast coverage and data acquisition through high-level options of Exploration planning and Informative planning;
- The Frontiers Priority Que (FPQ) that facilitates decision making for high-level planner options by storing information about the frontiers, which have an overlap with detected objects bounding box.

The remainder of the paper is organized as follows: Section II gives a detailed overview of the related work done in the scope of exploration planning and informative path planning (IPP). Section III overviews system and preliminaries. Section IV describes the proposed methodology in detail. Section V evaluates the proposed methodology, lists down preliminary results and comments on future work and, lastly, section VI concludes the paper.

II. RELATED WORK

The area of exploring and mapping unknown environments through mobile robots has received considerable attention in the recent past. Some of the state of the art approaches for mapping and exploration include frontier based approaches, information-theoretic approaches and adaptive sampling based approaches.

Frontier based approaches [9] for exploration are geometric in nature and explore the region by travelling to the boundaries between unknown and known regions. These boundaries are called frontiers. According to the original approach, during the exploration task, the closest frontier is chosen greedily as the next frontier to visit. There are a number of improvements made on this method. Instead of greedily selecting the closest frontier, [4] selects the next frontier that minimizes the velocity change to ensure maximum exploration speed. [10] introduced a method, which generated shorter exploration trajectories, by amalgamating a frontier based approach with local vector field strategy. In order to solve the problem of optimal coverage and fast exploration, [8] proposed the FUEL (Fast UAV Exploration) algorithm. This method proposed the Frontier Information Structure (FIS), which contains important information about the search space and is updated incrementally as exploration continues. By using FIS, they proposed a hierarchical 3-step planner for trajectory generation. The three steps are finding

global coverage paths, refining the local set of viewpoints, and generating minimum-time B-spline trajectories. This approach resulted in much faster exploration of the search area but did not prioritize information gain for the objects of interest, thus limiting its use for search and rescue operations.

Another approach, Adaptive sampling, requires random sampling of the search space to create viewpoints for exploration planning. [11] proposed an algorithm called Adaptive Search Space Coverage Path Planner (ASSCP) to generate a set of viewpoints by performing adaptive sampling that directs research towards areas with low accuracy and low coverage. [12] presented a new RRT*-inspired algorithm, which continuously expanded the single tree of candidate trajectories and refined intermediate paths. This method ensured global coverage and path utility function maximization. Some work involving a team of robots has also been explored. [13] proposed a method, using an adaptive sampling based approach, exploiting a team of Autonomous Underwater vehicles (AUVs) to explore the region. In this method, overall search space was partitioned in the regions close to each given AUV using voronoi diagrams and each robot runs adaptive sampling within its partition using map entropy of the environment. The environment used in this method has communication constraints and requires vehicles to initiate data sharing after some time. [14] further improved the partitioning procedure of search space by proposing voronoi partitioning which considers newly discovered obstacles and also updates regions continuously to improve load balancing between robots. The sampling-based approaches have shown state of the art results but they are computationally expensive which limits their usage for real world applications.

An alternative approach to sampling-based and frontier-based approaches include Information-Theoretic Planning. These methods normally optimize an information theoretic measure for exploration. [15] used a map entropy measure to select the next frontier to visit in a Frontier-based approach. [6] proposed method for information-theoretic planning approach, which chooses a trajectory from a set of global and local trajectories. Then they use gradient-based optimization to refine the chosen trajectory to maximize the Cauchy-Schwarz quadratic mutual information (CSQMI) objective. [16] proposed a method for target search problem which combines informative planning and obstacle awareness. They used layered optimization approach using Bayesian Optimization (BO) that balances the exploration-exploitation trade off between information gain, altitude dependent sensor performance, Field of View (FOV) and target re-observation. Another approach proposed by [7] solved the problem of exploration and informative planning by posing the problem as a correlated orienteering problem and travelling salesman problem. The proposed method provided anytime solutions in adaptive scenarios and also used a multiresolution sensor to gather target information.

In this paper, our approach is based on the work of [8] and add the ability in the structure to prioritize information gain for the detected objects of interest while maintaining a fast exploration rate. We propose Prioritized-FUEL, which

TABLE I
FRONTIER INFORMATION STRUCTURE OF THE CLUSTER

Data	Description
$Cell_i$	Frontier cells that belong to the cluster
$Cell_{avg,i}$	Average position of the cluster
BC_i	Bounding box of the cluster
BI_j	Bounding box of the object of interest
$Prob_{detect,j}$	Probability of the Detected Object
VP_i	Viewpoints around the cluster
$Cost_i$	Connection costs to other clusters

either selects a high-level exploration planner or informative planner based on information contained in Frontier Information Structure (FIS) and Frontier Priority Que (FPQ). Afterwards, both high level planners use common 3-step low-level planner to generate minimum-time trajectories.

III. PRELIMINARIES AND SYSTEM OVERVIEW

A. Frontier Information Structure

In frontier-based exploration [9], frontiers are defined as known-free voxel cells adjacent to the unknown cells. Clusters are defined as known-free voxel cells combined together. The method proposed by [8] introduced Frontier Information Structure (FIS) which provides richer and more organized information about the search space.

Whenever a new frontier Fr_i is detected, all the relevant information about that particular cluster is stored in the FIS using the cylindrical coordinate system. Table 1 summaries the data contained in the FIS. In our method, for the task of informative planning, we also add information about the object of interest to the FIS Structure. This includes the bounding box BI_j and the Probability of detection $Prob_{detect,j}$ of the j^{th} detected object. These two entries are used for decision making and will be explained later in the next section. When the map is updated, the information about the updated region is fetched and the bounding box $BB_{updated}$ is drawn around it. Afterwards it is checked if there is any overlap between the updated region bounding box $BB_{updated}$ and cluster bounding box BC_i . Similar to FUEL paper [8], for searching and clustering of new frontiers, we use region growing algorithm and then use Principal Component Analysis (PCA) to split each large cluster recursively in order to ensure robust decision making as large clusters do not help in characterizing different unknown regions.

B. Viewpoint Generation and Inter Frontier Cost Update

In our work, we use the methods proposed by [8] and [9] for the generation of viewpoints and for inter-frontier cost update. When a cluster Fr_i is created, the rich number of viewpoints $VP_i = \{x_{i,1}, x_{i,2}, \dots, x_{i,n_i}\}$ are generated so that the viewpoint with the maximum coverage can be selected through optimization. For each viewpoint, the information about the sampled point P_i , in cylindrical coordinate system, and its yaw angle ψ is stored i.e. $x_{i,j} = (P_{i,j}, \psi_{i,j})$. In addition to generating viewpoints, we also compute costs between frontier clusters. The connection cost is calculated as time

lower bound $t_{lb}(x_{k_1,j_1}, x_{k_2,j_2})$ between two viewpoints of clusters. The formula for calculating time lower

Algorithm 1 Prioritized FUEL

```

1: Initialize Frontier Information Structure FIS and Frontier Priority Que FPQ.
2: while Not whole region explored do
3:   Search for new frontiers  $fr_i$ 
4:   Generate Viewpoints and inter frontier cost using time-lower bounds eq.1
5:   if  $BC_i \cap BI_j$  then
6:     if  $Prob_{detect,j} > \epsilon$  then
7:       Append  $fr_i$  to FPQ
8:     while FPQ  $\neq \emptyset$  do
9:       InformativePlanner() // Information gathering
10:      end while
11:      end if
12:    else
13:      ExplorationPlanning() // faster coverage
14:    end if
15:  end while

```

bound is given in equation 1, where $P(p_{k_1,j_1}, p_{k_2,j_2})$ denotes collision free path between p_{k_1,j_1} and p_{k_2,j_2} , and v_{max} and ψ_{max} are velocity and yaw angle limits respectively. The collision free path is searched through A^* algorithm.

$$t_{lb}(x_{k_1,j_1}, x_{k_2,j_2}) = \max \left[\frac{\text{length}(P(p_{k_1,j_1}, p_{k_2,j_2}))}{v_{max}}, \frac{\min(|\psi_{k_1,j_1} - \psi_{k_2,j_2}|, 2\pi - |\psi_{k_1,j_1} - \psi_{k_2,j_2}|)}{\psi_{max}} \right] \quad (1)$$

IV. METHODOLOGY

In the proposed Prioritized-FUEL, we develop a hierarchical structure with two high-level options: Exploration planning and Informative planning. The overall structure is shown in Figure 1. The exploration planner is responsible for generating paths which result in faster coverage of the search area while informative path planner generates information-theoretic paths for gathering information about the detected objects. Both exploration and informative planners plan paths in three steps, which is similar to the original FUEL paper [8]. The detailed working of the Prioritized-FUEL is described in Algorithm 1 and explained below.

A. Detected Object Information and Frontier Priority Que

When new frontiers are searched, the information about all the clusters is maintained in the FIS. For the purpose of informative path planning, we also keep the information about detected objects, if any, in the FIS structure. According to the Table 1, we keep track of the bounding box BI_j and probability of the confidence of detection $Prob_{detect,j}$ of the detected object.

If and when the object is detected, the bounding box BI_j is drawn around it. Then precise checks are made

for all clusters and a list of all those clusters is returned whose bounding box BC_i intersects with the detected object bounding box. Afterwards, the probability of confidence of the detected object is checked and if it is greater than ϵ , then that cluster is added into the Frontier Priority Que (FPQ). Then the latter check is made to avoid the presence of false positives in detection.

B. Exploration vs Information theoretic Planning

Based on the status of the FPQ, different planners are activated accordingly. The only difference between informative planner and exploration planner is that in the former some of the clusters are given higher priority based on the possibility of the presence of an object of interest, while exploration planner does not prioritize any frontier but creates a minimum-time trajectory between any set of candidate clusters. The two different cases are mentioned here for further elaboration.

1) **FPQ contains at least one cluster:** If FPQ contains at least one cluster, the high-level option of exploration planning will pause and an informative planning option will be activated. In this case, clusters present in the FPQ are given higher priority and global paths are created to cover those clusters first. During this time, the UAV focuses on gathering data about the detected objects rather than exploring new areas. This might decrease the overall search time but ensures faster detection of the objects of interest, which can save lives in search and rescue operations. Here for data gathering, instead of circling around the object as in [7], we purely rely on local viewpoint refinement to select viewpoints which give maximum information of the object while not wasting energy circling around the object. The description of the working of the 3-step planner is given in the next subsection.

2) **FPQ contains no cluster:** If FPQ contains no cluster, then an exploration planner is chosen. Under the exploration planner, minimum-time trajectories are generated through all active clusters. This happens incrementally as new clusters are made along the search path. The purpose of the exploration planner is to ensure fast coverage of the search area. The detailed steps involved in generating minimum-time trajectories through a 3-step motion planner are mentioned in the next section.

C. 3-Step Planner

The 3-step planner is adopted from the original FUEL algorithm structure [8] and some modifications are made to incorporate our structure into it. The low-level planning procedure for both the high-level exploration planning option and informative path planning option is the same. The only difference is that during informative path planning, only selected or prioritized clusters are considered, while exploration planner considers all active clusters. The overall low-level 3-step planner includes Global Path Planning, Local Viewpoint refinement, and minimum-time B-spline trajectory generation. The overall structure can be shown in Figure 2.

1) **Global Path Planning:** Global planner creates a global path through the planner by posing the problem as the Asymmetric Travelling Salesman Problem, which creates an open-loop tour starting from the current viewpoint of

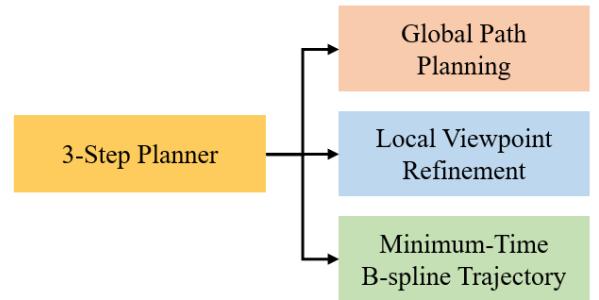


Fig. 2. The 3-step planner include Global Path Planning, Local Viewpoint Refinement and Minimum-time B-spline Trajectory Generation

cluster C_i and passing to all clusters. The cost from the current viewpoint x_0 to the x_k cluster can be evaluated using equation 2. Here time-lower bound $t_{lb}(x_0, x_{k,1})$ is used which was calculated and stored in FIS when frontiers were detected.

$$TSP_{cost}(x_0, x_k) = t_{lb}(x_0, x_{k,1}) + w_c \cdot c_c(x_{k,1}), \\ \text{where, } k \in \{1, 2, 3, \dots, N_{cluster}\} \quad (2)$$

In equation 2, $c_c(x_{k,1})$ is used as motion consistency cost which eliminates inconsistency by penalizing large changes in flight conditions. This inconsistency might rise due to several paths having similar time-lower bound.

2) **Local Viewpoint Refinement:** Here in this second step, the global path is improved based on the different viewpoints which were computed earlier. In original classical frontier-based approach, while calculating the global path, only a single viewpoint from each cluster is considered, which might not provide optimal collective coverage. For local viewpoint refinement, we create a graph of nodes from the current viewpoint x_0 to all viewpoints VP_i . In our method, we consider all active clusters and use the notation N_{at} to refer to them. After connecting nodes between clusters through directed edge, Dijkstra algorithm is used to search for the optimal local tour by minimizing the cost $LT_{cost}(x_0, x_{N_{at}})$ shown in equation 3. This approach is similar to some of the other proposed methods [8], [5], and [17].

$$LT_{cost}(x_0, x_{N_{at}}) = t_{lb}(x_0, x_{1,j_1}) + w_c \cdot c_c(x_{1,j_1}), \\ + t_{lb}(x_{N_{at},j_{N_{at}}}, x_{N_{at}+1,1}) + \sum_{k=1}^{N_{at}-1} t_{lb}(x_{k,j_k}, x_{k+1,j_{k+1}}) \quad (3)$$

3) **B-spline Trajectory generation:** For generating minimum-time B-spline trajectory, we use the method developed by [8] and [18]. The trajectory planner generates

smooth, safe, and dynamically feasible B-spline trajectories, and also optimizes all the parameters for the B-spline, which result in minimum-time trajectories.

The quad-rotor used during experiments is considered to be flat, so thus flat outputs include $x \in (p, \psi)$ where



Fig. 3. Here the UAV is shown ready for exploring the area. The light blue point cloud represents occupied region and colourful boundaries around occupied region represents frontier clusters. The red rectangle represents search area specified. The bounding box BI_j represents the bouding box around the object of interest. The grey shadows are the obstacles present in the environment.

$p \in (x, y, z)$. Thus the output can be shown as $X_{c,b} = \{x_{c,0}, x_{c,1}, \dots, x_{c,N_b}\}$, where $x_{c,i} = (p_{c,i}, \psi_{c,i})$ are the N_b+1 control points in p_d degree uniform B-spline. The knot number represents a number of control points used with a curve degree. The knot span here is referred to as Δt_b . The overall optimization problem can be written as equation 4 and we suggest the reader to refer to work of [8] and [18] for more details.

$$\arg \min_{X_{c,b}, \Delta t_b} fs + w_t T + \lambda_c f_c + \lambda_d (f_v + f_a) + \lambda_{bs} f_{bs} \quad (4)$$

Here in this equation, fs is the elastic band smoothness cost, R_s is the penalty matrix and f_c , f_v , f_a are penalties to ensure safe and dynamic feasibility and T is the total trajectory time. The detailed equations can be found in [8] and [18].

V. EXPERIMENTS

A. Scenario and Experiment Setup

In this section, we test the proposed Prioritized-FUEL algorithm in light simulator. The purpose of this experiment is to test the validity of the idea and its preliminary performance. The screenshot from the simulator is shown in Figure 3. The red boundary specifies search area, which needs to be explored. The black coloured bounding boxes BI_j represent bounding boxes which are drawn around objects of interest when detected. The grey color shadows represent unknown obstacles. Initially, UAV does not have any information about the presence of objects of interest or obstacles.

In our work, we assume that we have a perfect perception system, so that we can focus on improving the planning part of the system. This assumption helps us with two important

pieces of information. Firstly, the system knows the exact position of the object of interest so a perfect bounding box BI_j can be drawn around it, but UAV does not have any clue about the position of the object of interest at the start of the exploration process. It only comes to know about the presence of the object of interest when the bounding box

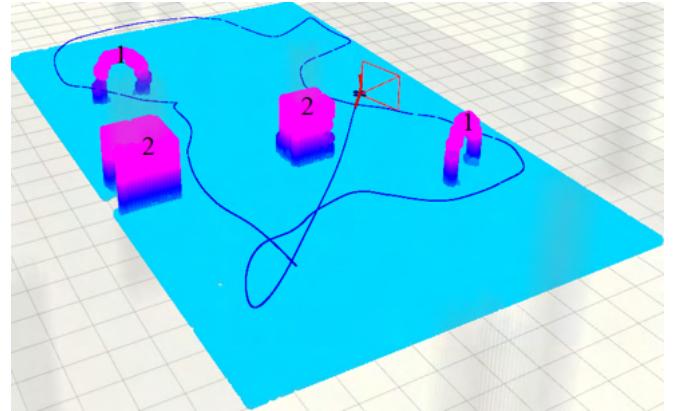


Fig. 4. The trajectory of the UAV is shown after fully exploring the region. The light blue point cloud represents the known region. objects with label "1" are objects of interest. Objects with label "2" are obstacles.

BI_j of the object of interest intersects with the bounding box BC_i of the frontier cluster. Secondly, we assume that $Prob_{detect,j}$ is always greater than ϵ , which is one of the requirements for informative planning as shown in algorithm 1. In experiments, with a focus on both perception and planning, bounding box BI_j of the object of interest and $Prob_{detect,j}$ would change as detection is carried out using some detection models [19], which might cause instability in the execution of planning routines. But in our system, this information is stable and reliable as we assume perfect perception system.

B. Results and Discussion

In order to evaluate the effectiveness of our proposed algorithm, Prioritized-FUEL, we compared it with the FUEL algorithm using the mentioned metrics over sample size of 10 experiments for each method:

- **Data Acquisition Time:** Time spent on exploring frontier clusters, whose bounding box BC_i have an overlap with the bounding box of the object of interest BI_j , once object of interest is detected;
- **Total Exploration Time:** Total time spent exploring the whole region.

The detailed results are shown in Table II. In our given simulation environment, Prioritized-FUEL was able to outperform the FUEL algorithm in the data acquisition part by focusing on exploring objects of interest first if detected, while keeping total exploration time almost the same.

As shown in Table II, the original FUEL algorithm spends on average 4.31 seconds on exploring the frontiers near objects of interest once the object is detected. While the proposed Prioritized-FUEL algorithm spends on average 2.38 seconds on exploring the frontiers near objects of interest.

TABLE II
COMPARISON OF FUEL AND PRIORITIZED-FUEL

Method	Data Acquisition Time (sec)				Total Exploration Time (sec)			
	Avg	Std	Min	Max	Avg	Std	Min	Max
FUEL [8]	4.31	1.38	1.79	9.00	55.50	2.99	49.21	59.43
Prioritized-FUEL	2.28	0.80	1.78	4.21	56.60	3.60	48.87	61.21

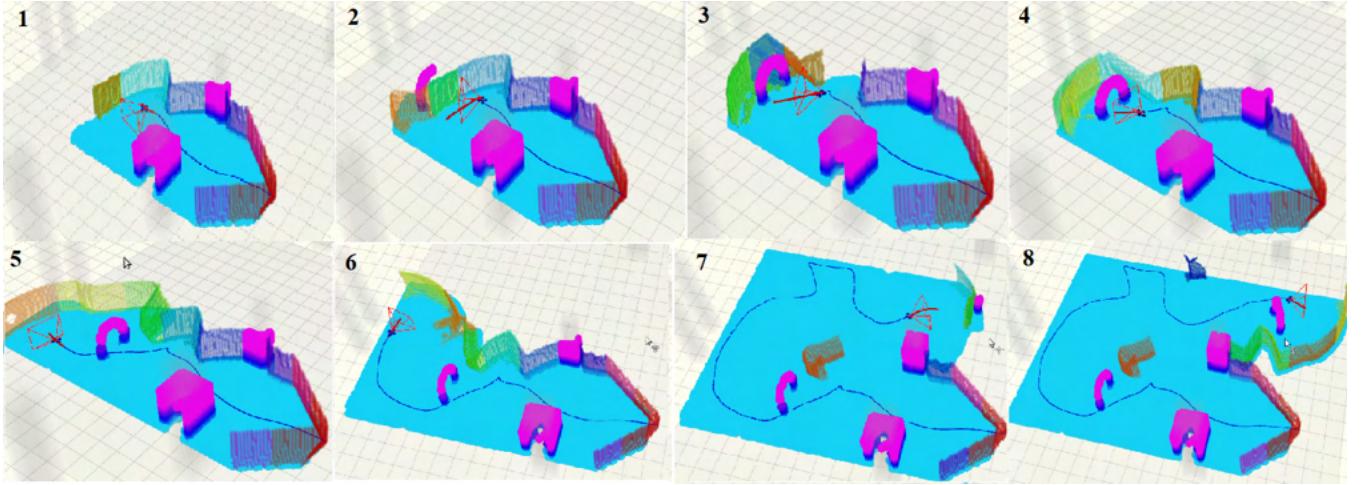


Fig. 5. The overall sequence of exploration is shown from one of the experiments from frames 1 to 8.

This is because in the original FUEL algorithm, UAV considers all frontiers as the same and generates minimum-time trajectories between them. This results in faster exploration of the environment. While in the Prioritized-FUEL algorithm, frontier clusters whose bounding box BC_i overlap with the object of interest BI_j are given higher priority and are added into Frontier Priority Que (FPQ). Thus UAV explores the high priority frontiers first by generating minimum-time trajectories between them, before exploring other normal frontiers.

Moreover it is shown in Table II that the total exploration time is not much affected. This is mainly because even while planning informative paths for data acquisition, minimum-time trajectories are generated and viewpoints are adjusted accordingly using Local Viewpoint refinement to maximize coverage. This shows that Prioritized-FUEL not only guarantees faster data acquisition but also faster exploration.

The sequence of the exploration from one of the experiments can be seen in Figure 5. The UAV can be seen to focus on objects of interest in frames 3 to 5 once it detects the first object of interest (circle) in frame 2. This can also be seen in frame 7 when the second object of interest (circle) is detected by checking the overlap between the bounding box of the cluster and of the circle. The final map is shown in Figure 4.

C. Future Work

The Proposed structure Prioritized-FUEL (Fast UAV Exploration) ensures a balance between faster coverage and informative planning. There are several more components which can be added to the structure to make it more robust. This includes altitude-aware data acquisition, multi-UAV

search and further testing in high fidelity simulations such as unreal engine and AirSim simulator.

Considering altitude while gathering information can help reduce uncertainty in sensor measurements. Obstacle-aware Adaptive Path Planning (OA-IPP) [12] incorporates altitude in it and shows that altitude-dependent sensor performance can be incorporated into cost or objective function. Moreover, Multi-UAV Search can help reduce overall search time by dividing search effort between multiple UAVs. One of the approaches famously used in the literature to divide the whole search region into partitions is voronoi partitions. One of the approaches proposed by [14] improved the partitioning procedure of search space by proposing voronoi partitioning which considers newly discovered obstacles and also updates regions continuously to improve load balancing between robots.

VI. CONCLUSION

In this paper, we proposed Prioritized-FUEL (Fast UAV Exploration) in order to ensure balance between fast exploration and data acquisition during reconnaissance operations. The hierarchical structure provides two high-level options: Exploration planning for faster coverage and Informative planning for data acquisition. In order to facilitate decision making for informative planner we modify Frontier Information Structure in the original FUEL paper to incorporate information about objects of interest. Moreover, we introduce Frontier Priority Que (FPQ) to store information about all the frontiers, which have a higher probability of the presence of the objects of interest near them. We test the proposed framework in a light UAV simulator and show that the prioritized-FUEL algorithm decreases the time to explore

objects' interest by almost 40% as compared to the original FUEL algorithm while keeping total exploration time of the search space almost the same. In future, we want to test and compare the algorithms in the AirSim simulator with the scenario of open sea reconnaissance operations.

ACKNOWLEDGMENT

This work is supported by the Robotics Institute Summer Scholars Program (RISS), Carnegie Mellon University, and The Hong Kong Polytechnic University. Also special thanks to Rohit Garg for sharing his experience with frontier-based exploration.

REFERENCES

- [1] S. Waharte and N. Trigoni, "Supporting search and rescue operations with uavs," in *2010 International Conference on Emerging Security Technologies*, 2010, pp. 142–147.
- [2] M. Korki, N. D. Shankar, R. Naymehshbhai Shah, S. M. Waseem, and S. Hodges, "Automatic fault detection of power lines using unmanned aerial vehicle (uav)," in *2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS)*, 2019, pp. 1–6.
- [3] P. Sujit, J. Sousa, and F. L. Pereira, "Uav and auvs coordination for ocean exploration," in *OCEANS 2009-EUROPE*, 2009, pp. 1–7.
- [4] T. Cieslewski, E. Kaufmann, and D. Scaramuzza, "Rapid exploration with multi-rotors: A frontier selection method for high speed flight," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2135–2142.
- [5] M. Selin, M. Tiger, D. Duberg, F. Heintz, and P. Jensfelt, "Efficient autonomous exploration planning of large-scale 3-d environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1699–1706, 2019.
- [6] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. Kumar, "Information-theoretic planning with trajectory optimization for dense 3d mapping," 07 2015.
- [7] S. Arora and S. Scherer, "Randomized algorithm for informative path planning with budget constraints," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 4997–5004.
- [8] B. Zhou, Y. Zhang, X. Chen, and S. Shen, "FUEL: fast UAV exploration using incremental frontier structure and hierarchical planning," *CoRR*, vol. abs/2010.11561, 2020. [Online]. Available: <https://arxiv.org/abs/2010.11561>
- [9] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation'*, 1997, pp. 146–151.
- [10] R. Shade and P. Newman, "Choosing where to go: Complete 3d exploration with stereo," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2806–2811.
- [11] R. Almadhoun, T. Taha, D. Gan, J. Dias, Y. Zweiri, and L. Seneviratne, "Coverage path planning with adaptive viewpoint sampling to construct 3d models of complex structures for the purpose of inspection," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7047–7054.
- [12] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto, "An efficient sampling-based method for online informative path planning in unknown environments," *IEEE Robotics and Automation Letters*, vol. 5, pp. 1500–1507, 2020.
- [13] S. Kemna, J. G. Rogers, C. Nieto-Granda, S. Young, and G. S. Sukhatme, "Multi-robot coordination through dynamic voronoi partitioning for informative adaptive sampling in communication-constrained environments," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2124–2130.
- [14] A. Dutta, A. Bhattacharya, O. P. Kreidl, A. Ghosh, and P. Dasgupta, "Multi-robot informative path planning in unknown environments through continuous region partitioning," *International Journal of Advanced Robotic Systems*, vol. 17, no. 6, p. 1729881420970461, 2020. [Online]. Available: <https://doi.org/10.1177/1729881420970461>
- [15] R. Rocha, J. Dias, and A. Carvalho, "Cooperative multi-robot systems a study of vision-based 3-d mapping using information theory," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 384–389.
- [16] A. A. Meera, M. Popović, A. Millane, and R. Siegwart, "Obstacle-aware adaptive informative path planning for uav-based target search," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 718–724.
- [17] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart, "Receding horizon "next-best-view" planner for 3d exploration," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1462–1468.
- [18] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen, "Robust and efficient quadrotor trajectory generation for fast autonomous flight," 2019.
- [19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.

Human Detection, Classification & Tracking in Context of Transit Systems

Chigozie Ofodike¹ and Christoph Mertz²

Abstract—With the advent and popularity of algorithms capable of detection-based tracking (DBT), one of its growing applications is in human detection and tracking (HDT). Leading research in HDT can be seen in surveillance systems, anomaly detection (e.g fall detection for senior citizens), and recently, social distance monitoring. In this paper, we present an application of contemporary HDT algorithms, on a real-time and ubiquitous entity—the mid-tech public transit bus system. All forms of DBT follow two innate steps: first, object detection, then association of the current object with the previous object. In the case of human detection, every instance of a detected human is then analyzed. In our project, we want to perform visual tracking from the transit bus. Each implementation is done by aggregating data (mainly pictorial) from cameras mounted on a bus with the Robotics Operating System (ROS) acting as the architecture supporting both the bus and the server structures. Our proposed system will allow for technological automations and implementations for human-specific observations.

Index Terms—keywords, Intelligent Transportation Systems, Surveillance Robotic Systems, Software, Middle-ware and Programming Environments, Computer Vision for Transportation

I. INTRODUCTION

With its wide use cases, algorithms involved in Human Detection and Tracking (HDT) have become relevant in recent years. Applications of these algorithms can be seen in surveillance systems, self-driving cars and even anomalous action detection in environments (i.e. a fall or a vandalism) [1]. The core steps of these algorithms are: informative region selection, feature extraction then classification (specifying person) [2]. Beyond these stages, most of the computation overhead occurs during the tracking stage [3]. Understanding that humans tend to be erratic, and often without a standardized shape, movement pattern or appearance, this overhead becomes more evident in cluttered environments or scenes with dense crowds [1], [4], [5]. Fortunately, many state-of-the-art (SOA) paradigms have suitable and efficient means to handle some of these resulting issues.

Our implementation is currently done on a single bus provided by Freedom Transit, a bus transportation service in Pennsylvania. Currently, in order for their analysts to review the recorded video of a bus' route, they must wait until the end of the day, when bus drivers are completely done with their respective routes. Then and only then are they able to gain access to it. After which analysts must sit and

¹Chigozie Ofodike is with Kean University, Union NJ, USA. Ofodikch@kean.edu

²Christoph Mertz with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA cmertz@andrew.cmu.edu

manually annotate the highly-repetitive and long streams of videos. This process is not only tediously time-consuming, it is prone to errors and it causes delays for timely information extraction. In this paper, we propose an implementation of real-time HDT to assist this system. Our proposed system will involve object detection and classification, to train a model to detect pedestrians. A classification mechanism to differentiate pedestrians from passengers either exiting or entering the bus.

The structure of this paper is as follows: In section II, we present our background study, showing related works, SOA object detection and tracking architectures. In section III, we present our system overview and our contributions to this system. Section IV will be our results and evaluations. V will be our conclusions and finally, VI will be our future work and discussions.

II. BACKGROUND STUDY

A. Related works

1) *General Object Detection*: Research in object detection and tracking (ODT) architectures have grown prevalent in recent times. Applications of ODT can be seen in medical imaging, automated robotics, image recognition and even surveillance systems [6], [7]. As mentioned in [1], [5] traditional object detection works by informative region selection, feature extraction, and classification. Traditional region selection is done with a sliding window approach. This method works by taking exhaustive sliding rectangular “patches” of fixed width and height for each image. Feature extraction then happens on each derived patch. After which a classifier is used to distinguish between objects in each frame. Due to the exhaustive nature required with the sliding windows, this traditional method is ineffective with real-time analysis [5], [8]. Nowadays, with the prevalence and utilization of Convolution Neural Networks (CNN) and deeply trained models, detections algorithms can occur at a much faster rate. We briefly discuss these SOA models in a later section.

2) *Object Detection on Embedded Devices*: Since these architectures need to be deployed to put to use, another popular research area is real-time object detection on embedded devices. Applications of object detection on embedded devices could be seen on autonomous vehicles, robotics, and C. Ye's BusEdge system [6]. Despite the recent successes of these algorithms, an issue common with the applications of object detection and tracking on embedded systems is the limited resources of these micro-systems. It becomes a question of which object detection architectures give us

the best tradeoff between speed and accuracy, while still effectively running on much lighter processors [9]. With the combination of significantly smaller processors and a computationally expensive algorithm, this is a difficult task. For one thing, when analyzing real-time data, [6] points out that most are unusable and very redundant. To that avail, there have been several successes that circumvent this issue. For example, Mobileye is a self-driving car company that implements a Road Experience Management system (REM). The company claims this system works by automatically uploading and processes anonymized data from cars already running its technology onboard [7]. Another company relevant in this space is RoadBotics. RoadBotics allows users to upload road-specific data to their server, then their deployed model handles the classification and analysis [10]. Specific to our needs, another leading approach facilitating this complex problem is NavLab's BusEdge system [6]. It employs an edge computing paradigm and an application called Auto-Detectron. Being that our system is an extension of his, we talk more about this in our "System Overview" section.

3) Pedestrian Detection and Tracking: Being that our project is fixated on how humans interact in relation to the transit system, our focus is strongly on pedestrian detection, tracking and observation. With the vast applications of human specific observations of object detections, there have been a multitude of implementations of SOA algorithms in this area. Zhang et al. [11] propose a method based on Tiny Yolo, training a model to detect the upper body of passengers entering/exiting a bus. Valastin et al. [12] show their different approaches to detection, tracking and crowd counting of pedestrians getting on and off a Metropolitan Train. They carry out their experiments in their reshapeable lab Pedestrian Accessibility Movement Environment Laboratory (PAMELA). This re-shapeable feature allows their lab to properly train pedestrian models no matter how rare, common, or messy of an event it is. They are able to handle problems we meet in our implementation such as properly detecting passenger flow (i.e. they are able to differentiate passengers entering from those exiting) and crowd counting of an active and crowded scene.

B. State of the Art Object Detection Architectures

1) Faster RCNN: Originally proposed by Ren et al. [13] RCNN, these architectures mainly consist of a layer of convolution neural networks (CNN) and Region Proposal Network (RPN). CNN is trained to extract appropriate features from the image, in this case features that appropriately describe humans. The RPN is a small neural network sliding on the last feature map of the convolution layer, predicting the existence of an object and the bounding box if an object is detected. The massive increase in analysis speed from 10 milliseconds per image to 2 milliseconds per image is heavily credited to this layer.

2) YOLO (You Only Look Once): Redmon et al. [14] propose a regression approach to object detection that requires only a single look at an image for object detection. It consists of 24 convolutional layers and two fully connected

layers and as the name suggests, it requires only one single forward propagation through the layers to detect objects. When compared to the architectures of RCNN, it tends to make more localization errors, but false positives are far less likely. In terms of speed of processing, YOLO's base model easily outperforms the already fast Faster RCNN—processing at 45 frames per second (fps) [15]. YOLO like RCNN comes with other versions, with its fast version processing at more than 150 fps. With such a massive processing rate, it is very suitable for live video processing with less than 25 milliseconds of latency [15].

3) H-YOLO: A Single-Shot Ship Detection Approach Based on Region of Interest Pre-selected Network: Although about ships, Tang et al. [16]. proposal of a single-shot detector on the pre-selected region of interest was the starting point of our passenger classifier. Using hue, saturation and value color space operations and a one-shot detector, they were able to extract pre-processed regions of interest at close to real-time.

4) Single Shot MultiBox Detector (SSD): Proposed by Liu et al. [17], SSD is a competing object detection model that works with a single phase analysis to detect multiple objects within the image. The SSD network is built on the VGG-16 model, where the feature map is extracted without the need of the bounding box proposals like that of RCNN. This map is then processed through six progressively decreasing convolution filters (the multi-box), generating. The use of multiple levels of filters allows about 8732 detections per object (class). The final layer, non-max suppression layer, eliminates the overlapping box by performing a bounding box regression effectively leaving the calculated final box with the highest overlap [17]. SSD's strength lies with its balance of ease of training speed and accuracy—being faster than both Faster RCNN and the base YOLO model, and more accurate than other single-stage methods.

C. State of the Art Tracking Models

1) Simple Online and Real-time Tracking with a Deep Association (Deep SORT): Deep SORT is the successor of SORT [18]. SORT is a high-performing two-stage tracking-by-detection framework that performs Kalman filtering in image space and data association using the Hungarian method. Put forth by Wojke et al. [18], Deep SORT builds upon its predecessor by adopting a deep association metric with recursive Kalman filtering on frame-by-frame data [5]. Although slower than its predecessor, the added association step drastically reduces the occurrence of identity switches among detected instances [18].

2) Joint Detection and Embedding (JDE): Proposed by Wang et al., unlike the two-stage tracking style of Deep SORT, JDE integrates the detector and embedding model into a single network. The combination of both stages removes the need for an additional layer of computation, therefore reducing the inference time [19]. Unfortunately, with this combination, these methods tend to be significantly less accurate than 2-stage methods, although capable of achieving near video rate inference [1].

3) *FairMOT*: A one-stage tracker that aims to balance speed and accuracy. Using an anchor-free detector, a heatmap, then a multiple-level data association step involving bounding box intersection over union, re-id features and Kalman filtering [20]. FairMOT's advantage comes from understanding that previous SOA's re-identification system is poor, due to its detection module being heavily favored over its re-id module. This work proposed by Zhang et al., aims to balance both modules. By implementing a multi-layer feature aggregation framework, its re-id module improves significantly.

III. SYSTEM OVERVIEW

Our detection pipeline will be deployed on the Gabriel BusEdge system as proposed and explained by C. Ye. [6]. The system's major components are hardware, early-discard filters, cognitive engines and finally sinks. Hardware on the bus are multiple wide angle cameras mounted at different positions on the bus, interior cameras, GPS, network antenna and an accelerometer (see Figures 1 and 2). The early-discard filters are Ye's lightweight preprocessing mechanism, and we choose SSD with MobileNet. This allows ad hoc analysis instead of analysis on highly-repetitive data effectively promoting scalability. Cognitive engines are the more computationally heavy computer vision models that handle and analyze distilled data from the bus, YOLO is our choice here. Finally the sinks, which represent the final component used for data analytics and visualizations. Our choice here is Christensen's proposed LiveMap system [21]. The complete system uses Robotics Operating System (ROS) as its base architecture. See [6] for a more detailed and thorough description for the Gabriel BusEdge system.

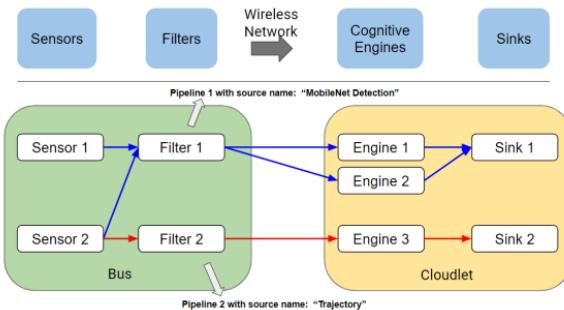


Fig. 1: An abstraction of C Ye's BusEdge pipeline.



Fig. 2: Pictures of the Hardware. (From left to right: bus computer; exterior camera; interior camera; GPS and network antenna.)

A. Our Contributions

Our implementation is centered around aggregating data for human activity around the transit system. Although at

the beginning stages, we implement a human detection model, then a classification module capable of differentiating pedestrians around from those boarding on a specific transit route. Being that humans are the focal point of public transit, this creates a solid starting point for future automations.

B. Proposed Methodology for our Pedestrian Detection and Passenger Classification.

- 1) Grab and read a frame.
- 2) Apply a fine-tuned object detection model to each individual frame. To only detect people, we simply discard the information of every other class.
- 3) Get and store bounding boxes, scores and labels of each instance of each detected human.
- 4) Select only pedestrians above a .70 confidence score—this way we eliminate false positives such as mannequins and road signs at a distance.
- 5) Instantiate a passenger counter.
- 6) Create an invisible bounding box (we call this our boarding zone) where passengers must enter or exit.
- 7) For each properly identified pedestrian instance, if the instance crosses the boarding zone, we compute the intersection over union (IoU) of our zone against the detected pedestrian.
- 8) If the resulting confidence score is greater than .50, this person instance is treated as a passenger.

IV. RESULTS AND EVALUATIONS

1) *Results*: Although promising, there are still some challenges with our pipeline. First off, as shown in Fig 3, there tends to be some mislabelling of pedestrian instances. Instances such as this, and accidental mannequin selections occur at a confidence threshold of .75. Increasing the threshold further would cause the model to ignore instances in darkened scenes (such as that in Fig 4). Another common issue that came up are pedestrian instances missed because of slight occlusions (shown in Fig 5). This specific instance is barely covered by the obstructing sign, yet it was completely ignored. This same instance in the frame before commanded a .90 confidence score. This can easily be fixed by either implementing a SOA tracker, allowing preservation of the identity of each instance between each frame change, even behind partial obstructions or the re-identification of instances even through complete obstruction.



Fig. 3: Sign to the extreme right mislabelled as a person.

2) *Evaluations*: Evaluating our logic on unseen data, we see some issues. Our pedestrian detector seems to get all pedestrians in a given scene, but it fails to differentiate between a reflection and the actual passenger as seen in Fig. 6b. This was an unanticipated event, as it is the first time such a detection occurred. Being that the passenger



(a) Original frame.



(b) Object detection.



(c) Fine-tuning for pedestrians.



(d) Passenger specification.

Fig. 4: Successful pedestrian detector and passenger selection.

classification module only cares about the instance in the invisible boarding zone, this still seems like an effective method of passenger classification.

V. CONCLUSIONS

In this paper, we proposed an addition to Freedom Transit's current analysis pipeline. With the help of SOA algorithms, and C Ye's BusEgde system as the architecture behind this, we were able to effectively detect humans around specific bus routes. The research and work done here provides a foundation to assist Freedom transit with their data analysis processes.

VI. FUTURE AND DISCUSSIONS

Future plans will be to use already existing benchmarks such as those provided by [22], [11] and [23]. [22] works well with detecting occluded instances in urban areas, whereas [11] is a significantly more diverse pedestrian dataset. [23] claims to have a strong generalization ability. Evaluation of these datasets, recommended training paradigms and more were presented well by Hasan et al. [2]. Another implementation we plan on is for people with disabilities. Rarely are they adequately represented in popular pedestrian datasets. Then, a tracker for all instances. Giving our system the ability to tell detected humans apart, effectively establishing metrics for possible analysis. A counter-



(a) Pedestrian on the extreme left will be missed in the next frame.



(b) Instance missed due to occlusion.

Fig. 5: Interesting, yet correctable misses.



(a) Original frame.



(b) Person detection evaluation.



(c) Passenger specification evaluation.

Fig. 6: Evaluation of pedestrian detector and passenger selection.

intuitive, yet interesting notion put forth by [2] is that general object detection models tend to work better than current SOA pedestrian-specific detectors on new and population-dense scenes. We will analyze and compare SOA pedestrian-detectors, SOA general object detection models, and our pedestrian-specific model, specific to the Pittsburgh area to see how each fares and which is best for our specific needs. To correct the issue with detection reflections as humans, being that having the bus in the image is not necessary for any of our analysis, we will simply only perform our detection in a section of each frame without the bus in it.

ACKNOWLEDGMENTS

This research is based upon work supported by the National Science Foundation under Grant No. 1659774. We thank the Robotics Institute Summer Scholars program for making this experience possible, Rachel Burcin and John Dolan for their support and guidance throughout the program and Albert Ye and Tom Bu the master students we worked under.

REFERENCES

- [1] M. Paul, S. Haque, and S. Chakraborty, "Human detection in surveillance videos and its applications: a review," *Eurasip Journal on Applied Signal Processing*, vol. 176, pp. 1–16, 2013.
- [2] I. Hasan, S. Liao, J. Li, S. U. Akram, and L. Shao, "Generalizable pedestrian detection: The elephant in the room," 2020.
- [3] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, X. Zhao, and T.-K. Kim, "Multiple object tracking: A literature review," 2017.
- [4] M. E. Hussein, W. Abd-Almageed, Y. Ran, and L. Davis, "Real-time human detection, tracking, and verification in uncontrolled camera motion environments," *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, pp. 41–41, 2006.
- [5] Z.-Q. Zhao, P. Zheng, S. tao Xu, and X. Wu, "Object detection with deep learning: A review," 2019.
- [6] C. Ye, "Busedge: Efficient live video analytics for transit buses via edge computing," Master's thesis, 2021.
- [7] "Autonomous driving amp; adas (advanced driver assistance systems)." [Online]. Available: <https://www.mobileye.com/>
- [8] J. Lee, J. Bang, and S.-I. Yang, "Object detection with sliding window in images including multiple similar objects," in *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, 2017, pp. 803–806.
- [9] M. Afifi, Y. Ali, K. Amer, M. Shaker, and M. Elhelw, "Robust real-time pedestrian detection on embedded devices," 2020.
- [10] A. Quillen, "Roadbotics: Make data-driven decisions," Aug 2021. [Online]. Available: <https://www.roadbotics.com/>
- [11] S. Zhang, Y. Xie, J. Wan, H. Xia, S. Z. Li, and G. Guo, "Widerperson: A diverse dataset for dense pedestrian detection in the wild," *CoRR*, vol. abs/1909.12118, 2019. [Online]. Available: <http://arxiv.org/abs/1909.12118>
- [12] S. A. Velastin, R. Fernández, J. E. Espinosa, and A. Bay, "Detecting, tracking and counting people getting on/off a metropolitan train using a standard video camera," *Sensors*, vol. 20, no. 21, 2020. [Online]. Available: <https://www.mdpi.com/1424-8220/20/21/6251>
- [13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2016.
- [14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.
- [15] ——, "You only look once: Unified, real-time object detection," 2016.
- [16] G. Tang, S. Liu, I. Fujino, C. Claramunt, Y. Wang, and S. Men, "H-YOLO: A Single-Shot Ship Detection Approach Based on Region of Interest Preselected Network," *Remote Sensing*, vol. 12, no. 24, p. 4192, 2020. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-03096535>
- [17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," *Lecture Notes in Computer Science*, p. 21–37, 2016.
- [18] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," 2017.
- [19] Z. Wang, L. Zheng, Y. Liu, Y. Li, and S. Wang, "Towards real-time multi-object tracking," 2020.
- [20] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "Fairmot: On the fairness of detection and re-identification in multiple object tracking," 2020.
- [21] K. Christensen, "Computer vision for live map updates," Master's thesis, 2019.
- [22] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 304–311.
- [23] S. Zhang, R. Benenson, and B. Schiele, "Citypersons: A diverse dataset for pedestrian detection," 2017.
- [24] H. Qu, M. Wang, C. Zhang, and Y. Wei, "A study on faster r-cnn-based subway pedestrian detection with ace enhancement," *Algorithms*, vol. 11, p. 192, 11 2018.
- [25] S. Velastin, R. Fernández, J. Espinosa, and A. Bay, "Detecting, tracking and counting people getting on/off a metropolitan train using a standard video camera," *Sensors (Basel, Switzerland)*, vol. 20, 2020.
- [26] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016.
- [27] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," 2019.
- [28] E. M. Silva Machado, I. Carrillo, M. Collado, and L. Chen, "Visual attention-based object detection in cluttered environments," in *2019 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computing, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, 2019, pp. 133–139.
- [29] S. Zhang, Y. Wu, C. Men, N. Ren, and X. Li, "Channel compression optimization oriented bus passenger object detection," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–11, 03 2020.
- [30] Y. Wang, K. Kitani, and X. Weng, "Joint object detection and multi-object tracking with graph neural networks," 2021.

Control System Modeling for Closed-Loop Controlled Ventilator using MATLAB Simulink

Zaria Oliver¹ Lu Li² and Howie Choset²

Abstract—Made more apparent due to the ongoing COVID-19 global pandemic, is the lack of efficient ventilators ready to provide aid to the numerous lives impacted by widespread disease. Intensive care unit (ICU) ventilators are sizable and costly, while existing low-cost, portable ventilators require further development to tackle treatment of acute respiratory distress syndrome (ARDS) injuries including COVID-19. The RoboVentilator (RoboVent) is a compact, low-cost mechanical ventilator whose performance is comparable to that of sophisticated ICU ventilators. The device combines mechanical and pneumatic design with robotic control. This project develops a model for the RoboVent closed-loop control system to regulate behavior of the actuation and sensing components. By using Simulink, a graphical modeling program through MATLAB, a control model can be created and developed allowing for adjustment of RoboVent parameters. This includes values such as respiratory rate, tidal volume, and set pressure. The contribution of this work is a control model permitting for system validation and digital twins virtual testing to be performed without heavy modification of the RoboVent hardware. Specifically, this paper focuses on the overall ventilator layout and the gas mixer component of the RoboVent.

Index Terms—Medical Robots and Systems, Human Centered Robotics, Control Architectures

I. INTRODUCTION

Ventilators are medical devices which fully provide or assist in breathing for patients with respiratory distress [1]. When patients are unable to breathe independently, a ventilator is used to consistently force air into the lungs with positive pressure [2] for the needed period of time. A 2020 study referenced by the National Institutes of Health (NIH), conveys the urgent need for efficient ventilators to address the large demand created by widespread disease [3]. 56 percent of COVID-19 ICU patients required mechanical ventilation at a hospital in Wuhan China. 76 percent of those patients required invasive mechanical ventilation, which is ventilation in which an endotracheal tube is used to deliver air to a patient. 79 and 86 percent were the mortality rates of patients relying on non-invasive versus invasive mechanical ventilation. Many COVID-19 ICU patients have an increased chance of developing ARDS, a fatal lung injury involving fluid leakage into the lungs [4]. Mechanical ventilation is necessary for supporting ARDS patients through recovery. Shockingly, by March of 2020, there was a shortage of

ventilators for sale from medical equipment providers [5]. During a global pandemic such as COVID-19, the high demand, high cost, and low availability of ventilators is a recipe for disaster.

A. About the Ventilator

The RoboVentilator is a compact, low-cost ventilator with advanced functionality and rapid deploy ability. Not only is its performance equivalent to sophisticated level ventilators, it is also more easily producible, an important feature to have in preparation for possible future disease outbreaks. Current low cost ventilators are not advanced enough for COVID-19 complications. The ventilator allows for parameter customization, providing a specifically catered device for the patient's particular needs. These parameters include: tidal volume, set pressure, respiratory rate, and positive end-expiratory pressure (PEEP). Averaging 400-500 mL in a healthy person [6], tidal volume is the amount of air moving into and out of the lungs per breath. Respiratory rate refers to the amount of breaths per minute a person takes [7]. A fascinating parameter is the positive end-expiratory pressure (PEEP), a pressure that maintains airway pressure upon exhalation. This prevents collapsing of air sacs within the lungs of ARDS patients. The RoboVent parameters are tested based on current COVID-19 ICU patient data such as O₂ saturation measurements, respiratory rates, CO₂ levels, toxin measurements and more.

The RoboVent, shown in Figures 2 and 3, is a closed-loop system that allows for processes to be carried out to desired states without human interaction.

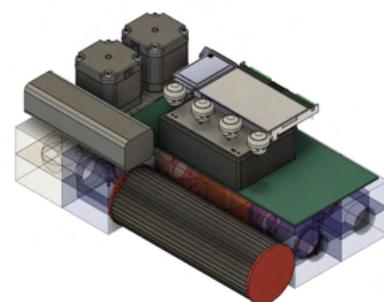


Fig. 1. RoboVentilator Computer Aided Design

B. RoboVent Design

The device's design, displayed in Figure 3, starts at the oxygen and air gas sources of the ICU environment. The

¹Zaria Oliver is with the Department of Mechanical Engineering, University of Maryland, Baltimore County, Baltimore, MD, 21250 zoliver1@umbc.edu

²Lu Li and Dr. Howie Choset are with the Biorobotics Lab, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 liliu12@andrew.cmu.edu and choset@andrew.cmu.edu



Fig. 2. RoboVentilator Device with Digital Monitor Displaying Pressure, Flow, and Volume

inspiratory section of the system deals with the patient's breath in, while the expiratory components have to do with the breath out. Oxygen is sourced from an oxygen cylinder, most commonly of size E (660L) [8] at 50psi. The air supply is compressed external air at a pressure of around 58psi [9]. Each of the gas streams runs through a corresponding valve allowing for inspiratory flow rate adjustment based on data from a flow sensor down-the-line of the device path. These streams join a gas blender which mixes the gasses and outputs a deliberate flow of mixed gas at a pressure based on prior values. The tube following this output leads to the patient. However, along this route is an emergency breathing valve, an inspiratory flow sensor, an inspiratory pressure sensor, and a check valve. These components allow for a patient to breathe external air if needed, relay flow and pressure information to other components of the device and the medical professional, and ensure that air is flowing in the correct direction. Should the pressure through the emergency breathing valve exceed a set value at the inlet, the valve will open and remain open until the measured pressure is again within the acceptable range. The check valve measures the input pressure compared to the outlet pressure. A negative pressure read across the valve indicates that the inspiratory airflow is going in the wrong direction, away from the patient. The process described thus far encompasses the inspiratory section of the ventilator. The expiratory section starts at the patient, and also moves through a corresponding check valve which makes sure the exhaled air is flowing away from the patient. The expiratory tube is also met with a pressure sensor and flow sensor, where the pressure sensor not only reads general expiratory pressure values, but also provides information to the emergency pressure release valve. This valve is essential in relieving excess pressure preventing a possible build-up in the lungs.

In order to alter values for testing the main parameters of the device, hardware must be modified. This inconvenience is why a model that allows for hands-off testing of the computer system, parameters, and digital representation of the physical processes is such a valuable tool to have. The objective of this project is to: 1. Produce a general layout of the RoboVent system using the MATLAB block library tools. 2. Create a gas mixer model to replicate the O₂/Air source gas-blending

subsystem. The models created in this project are novel and do not yet exist in the current Simulink library.

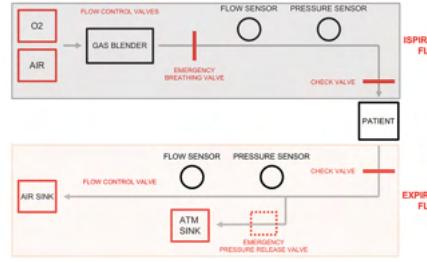


Fig. 3. Flow Diagram Representing General RoboVentilator Design

II. METHODS

The MATLAB Simulink graphical programming environment was used for modeling the ventilator due to its ability to model a nonlinear system and have set initial conditions. This resource is beneficial to a design team, as it allows for multiple domain environments [10] where subsystems and their overall arrangement can be modeled. The block diagramming tool enables system modeling in an organized fashion and also permits parameter setting of that correlating to each block tool. This allows for the models created to be easily understood by users and for specific conditions to be set. To create the more complex gas system models for the RoboVent system, the block tools were integrated together. Block functions used for this project included mathematical tools, signal type converter blocks, and gas system specific instruments. To guide the creation of the model using Simulink tools, based on the general layout of the RoboVent design, a MATLAB Simulink example medical ventilator model [11] was referenced.

III. MATHEMATICAL MODEL

A. Gas Mixer Subsystem

As shown in Figure 3, oxygen and compressed air each run through a proportional valve providing a change in output flows and pressures. The two gas streams are then combined in a gas blender system. Simulink's gas chamber specific block tools were not intricate enough to model this mixing apparatus. The program's constant volume chamber block does not allow for an outlet stream, and the controlled reservoir block assumes that volume is infinite which is not true for the gas blender. A new model was needed in order to represent this mixture of input gas. The following math model was used to provide basis for the gas mixer block diagram:

Ideal Gas Equation

$$PV = nRT \quad (1)$$

Where P is pressure in Pascals (Pa), V is volume in liters (L), n is the number of moles, R is the gas constant

$$Volume = \int_{t_1}^{t_2} \dot{V} dt \quad (2)$$

in liter-Pascals per Kelvin-mole (L·Pa·K⁻¹·mol⁻¹), and T is temperature in Kelvin (K). It was assumed that the gas particles move randomly and are therefore ideal. The fact that volume is the integral of flow rate over time was important in the gas mixer modeling process.

IV. MATLAB SIMULINK MODEL

It is important to note that all purple wires in the following models represent ventilation tubes, and that the symbols A and B associated with each block represent the input and the output of the respective block tool.

A. RoboVent System Model

A gas source block starts off the inspiratory section of the model shown in Figure 3, representing a mixture of oxygen and air. A pressure source block connects to this path and contains the output value from the gas mixer subsystem. The value emerging from this block is the value for the pressure of the gas mixture based on calculations conducted with Equations 1 and 2. The pressure source adjusts the previous pressure value of the data flowing from the gas source block. An inspiratory flow valve following this path was represented by a gas local restriction block in Simulink. This block allows for changes in a restriction area, the size of the space through which air can flow. The local restriction was implemented to control flow rate of the gas stream. A larger restriction area allows for more gas molecules to pass through the tube and therefore passes through more liters of air per minute. A smaller restriction area does the opposite, resulting in a lower flow rate. To obtain maximum flow rate, the restriction area can be set to the equivalent area of a 15 mm inner diameter tube used for ventilation. The tube component is a pipe block and simply represents the inspiratory tube. While the pipe allows for heat transfer, thermal components, like humidity, were not focused on for this project. The pressure and flow sensors attached to the path measure a pressure and flow rate values at the connection point. These sensors are built in Simulink blocks, and both connect to a scope. The scope is a graphical representation of the sensor readings. The connection from the sensors to their respective scopes contain PS-Simulink converters, which convert a physical signal (PS) to a Simulink signal. The final component of the inspiration process is the check valve subsystem. The inspiratory and expiratory check valve subsystems consist of their own models, which were not focused on for this project. The inspiratory tube is met with the input source of a gas chamber block representing the patient. The gas chamber volume is adjustable and represents the lung volume, and the area of the inlet and outlet can also be adjusted. This model allows for direct adjustment of the gas flow rates and

the restriction area of the inspiratory tube, and provides a user with the inspiratory pressure and flow rate values as well as the pressure of the mixed gas.

The expiratory model starts with the gas chamber representing the patient. Just like the inspiratory tube, a pressure and flow sensor is attached along with the corresponding scopes. The scopes will now read the pressure and flow rate values of exhalation. The emergency pressure check subsystem follows the sensors. Once again, this subsystem depends on the pressure sensor value and was also not focused on for this project. However, the pressure check outputs lead to two different paths. The flow is directed to another local restriction valve for flow adjustment, and released into an air sink modeled by a gas reservoir; however, pressure that exists in excess is directed to an atmosphere sink.

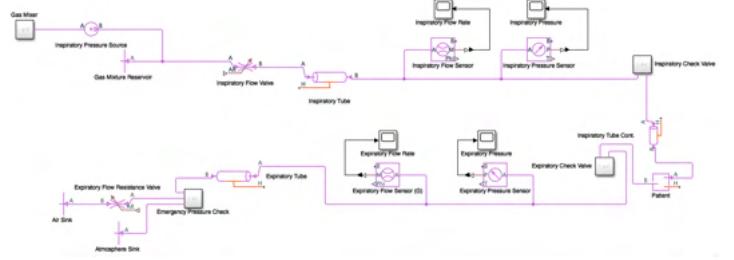


Fig. 4. MATLAB Model of RoboVent System

B. Gas Mixer Subsystem Model

The gas mixer model begins with two constant value blocks representing the flow rates of oxygen and air. These values and their simulation times are adjustable, and in Figure 5, the average flow of 60 liters per minute (LPM) [12] for an infinite simulation time is used for each flow. The flow values are connected to integrator blocks that take the integral of the input signal. Mathematically, this represents the integral of the volumetric flow rate which is volume, shown in Equation 2. The volume of oxygen and air is necessary in order to calculate the number of moles of each gas using the ideal gas equation shown in Equation 1. Scope blocks are connected to the outputs of each constant and integrator block, providing the user with a graphical model of the volumetric flow rate versus the volume over the simulation time. The MATLAB Function blocks allow for the input signal to run through a code integrated into the block. The function blocks connected to the integrator outputs each contain a MATLAB code for the ideal gas equation, where the number of moles of the respective gas is found given the volume, pressure, temperature, and gas constant. The value of the gas constant used for each block was 8314 L·Pa·K⁻¹·mol⁻¹. Room temperature was used for each function as well, since this is the environment that exists within the hospital ICU. Pressure values were set to 50 and 58 psi for oxygen and air respectively, as mentioned in the RoboVent design. The pressure values were converted to Pascals in order to keep units constant. The output signal values for the number

of moles of oxygen and air meet at a sum block. This block allows for an adjustment of the number of inputs and whether each input is a positive or negative value. The number of moles was summed in order to obtain the total number of moles of a mixture of the two gases. The summed value is then entered into another MATLAB function block which also contains the code for the ideal gas equation. However, this block has two inputs, one for the value n of the mixed gas and another for the volume once again. Given the values of T , n , R , and V , an output is provided for the pressure of the mixed gas. This output leads to a display block which simply shows the value outputted in Pascals. The output is also connected to a scope block for graphical modeling of the pressure over the simulation time. This subsystem exists within the larger RoboVent model. Specifically, the overall output of the gas mixer model is the input for the RoboVent model pressure source. The ability to modify the flow rates and the run time for this subsystem allows for alteration of the set pressure and tidal volume parameters.

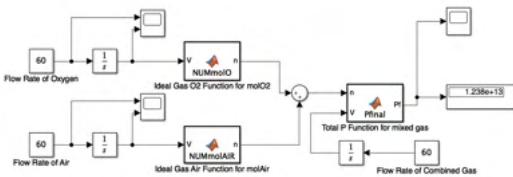


Fig. 5. MATLAB Model of Gas Mixer Subsystem

V. DISCUSSION

The collaboration of the model components allow for various values for flow and pressures to be tested. These values affect the parameters of tidal volume and respiratory rate. The gas mixer subsystem and flow valves, specifically, affect the set pressure parameter. Also, the emergency pressure check subsystem works along with the model's customizability to allow for testing of the PEEP parameter. These models are useful tools in device testing to ensure that the Robot Ventilator performance is comparable or superior to a sophisticated ICU device. Also, developing a control model for the RoboVentilator allows for the design and physical system to be updated with components that have not yet been included. Specifically, the gas mixer focused model allows for flow rates to be adjusted prior to mixing, which the current design layout does not include. Not only does this project provide models that can be added to the MATLAB specific and various other forums, but it also highlights the need for further development of more complex Simulink modeling tools. Future work for this project includes development of detailed models for the remaining check and emergency valves. It is also proposed that addition of a mass flow rate block, instead of a local restriction block in the overall model, would be a better tool for flow rate adjustment. Overall, this research brings us a step closer to ensuring efficient ventilators for future ARDS patients and strengthens our emergency preparedness for future global disease outbreaks.

ACKNOWLEDGMENT

Thank you to the RISS program and program managers, Rachel Burcin and John Dolan, for allowing me to partake in a wonderful opportunity. I would also like to thank Lu Li, Dr. Choset, and the rest of the Biorobotics lab for providing me with support as I conducted research this summer.

REFERENCES

- [1] G. D. Baura, *Mechanical Ventilators*, 1st ed. California: Elsevier, 2012, pp. 217–235.
- [2] C. Macmillan, “Ventilators and covid-19: What you need to know,” 2020.
- [3] K. Iyengar, S. Bahl, R. Vaishya, and A. Vaish, “Challenges and solutions in meeting up the urgent requirement of ventilators for covid-19 patients.”
- [4] A. L. Association, “Acute respiratory distress syndrome (ards).”
- [5] W. P. King, J. Amos, and M. A. et al., “Emergency ventilator for covid-19,” 2020.
- [6] S. Hallett, F. Toro, and J. V. Ashurst., “Physiology, tidal volume,” 2021.
- [7] C. Clinic, “Vital signs.”
- [8] D. King, B. Houseman, and M. Decker, “Gas cylinders,” 2021.
- [9] S. Das, S. Chattopadhyay, and P. Bose, “The anaesthesia gas supply system,” *Indian journal of anaesthesia*, vol. 57(5), pp. 489–499, 2013.
- [10] MathWorks, “Model-based design: From concept to code.”
- [11] ———, “Mechanical ventilator with lung model.”
- [12] A. Amitai and R. H. Sinert, “What is the setting for inspiratory flow rates in mechanical ventilation?”

Model-Based Reinforcement Learning for Off-Road Navigation

Ashley Peake¹, Samuel Triest², Wenshan Wang², and Sebastian Scherer²

Abstract— Autonomous off-road driving is an important extension of unmanned vehicle technology for applications in exploration, search and rescue, and construction. In any off-road driving task, vehicles must be able to navigate within cluttered, dynamic environments and interact with various obstacles. Because the dynamic properties of the environment are critical for successful navigation, semantic and geometric features alone are insufficient for planning. However, explicitly modeling the dynamics is difficult due to the size and complexity of such environments. To overcome this, we use model-based reinforcement learning (MBRL) to learn a physics model of the environment that can be incorporated into planning. We train our model by extending the MuZero framework developed by DeepMind [1] and use a Monte Carlo Tree Search for planning. Furthermore, we incorporate multimodal sensor information into our model by learning a robust latent space representation. We evaluate the success and efficiency of this method in several simulated environments.

Index Terms— Reinforcement Learning, Autonomous Vehicle Navigation, Machine Learning for Robot Control

I. INTRODUCTION

Recent developments in deep learning and robotic control have opened a host of possibilities for autonomous driving [2], [3]. These autonomous systems utilize on-board sensors to extract information about the structure of their environments to make safe and efficient driving decisions. Although often considered in the context of traditional driving applications, these developments are also relevant for off-road driving. Autonomous off-road driving is important for applications in search and rescue and ground exploration. In these tasks, vehicles must navigate unknown and dynamic environments, often without roads or predefined traffic patterns. The time-sensitive and potentially dangerous nature of these tasks makes autonomous vehicles a particularly beneficial option. It is thus critical to develop intelligent, efficient, and safe planning algorithms for unmanned ground vehicles.

There are, however, significant challenges to overcome. Off-road environments are generally dynamic and cluttered with obstacles. By definition, they lack roads, but in many cases these environments also lack visually distinct paths to follow. As such, the semantic and geometric features alone are insufficient for planning; the dynamic properties of the environment must be considered. However, the size and complexity of such environments makes the dynamics difficult to explicitly model. Even when possible to construct, such an analytic model would be time consuming to build and brittle to environmental changes.

To overcome this, we propose a model-based reinforcement learning (MBRL) approach for off-road driving. MBRL is a common method for learning optimal behavior in complex, unknown environments. In a MBRL approach, an agent collects experience from the environment to learn a dynamics model (i.e. how the environment responds to different actions). In this way, we avoid the need to have a completed, pre-determined model of the environment. The learned model can then be efficiently used for planning. In this paper, we present an MBRL method that makes the following contributions:

- We propose an architecture for learning environment dynamics relevant for planning.
- We design a latent space to encode state information.
- We use Monte Carlo Tree Search with the learned model to select actions for optimal vehicle control. This is an extension of the MuZero framework developed by DeepMind [1], and we test it in several simulated environments.

The remainder of this paper is organized as follows. In Section II, we discuss related approaches to the off-road driving problem. In Section III, we introduce the necessary reinforcement learning background. In Section IV, we discuss the details of our approach. In Section V, we present preliminary results for two simulated environments. Finally, in Section VI, we detail plans for future work.

II. RELATED WORK

In previous work, model-free reinforcement learning methods have been proposed for autonomous off-road driving [4]. Unlike MBRL, this approach involves directly learning optimal behavior by collecting experience from environment. However, the interactions between agent and environment are ultimately inefficient for planning in an off-road setting. Furthermore, the environment dynamics are never explicitly modeled, meaning they cannot be exploited in downstream navigation tasks.

Bajracharya et al. propose an end-to-end learning framework for off-road driving that employs the use of classifiers to determine and map terrain traversability [5]. These classifiers extract geometric features from the environment to predict traversability. This information is subsequently used for path planning. In this case, model predictions are limited strictly to the environment's geometry. In complex, dynamic environments, geometric features alone are insufficient for optimal navigation. Our approach follows a similar framework by planning based on a learned model, but we allow the model to learn the relevant dynamic properties of the environment.

¹Department of Applied Mathematics at Wake Forest University

²Robotics Institute at Carnegie Mellon University

Rhinehart et al. devised a deep imitative model approach for autonomous driving [6]. This method combines MBRL with imitation learning – i.e. learning from expert demonstration. This combination maintains the benefits of MBRL without relying on reward function crafting. However, this method still requires sufficient expert demonstration. Furthermore, it is tested only in on-road driving tasks. Off-road driving requires more adaptability and generalizability at test time than imitation learning offers.

MBRL has been previously successful in autonomous driving applications. Wu et al. propose a Dyna-style MBRL approach [7]. This method uses a learned dynamics model to forward simulate state transitions. These transitions are stored as experience to train a reinforcement learning model to select actions. This is significantly more complex and computationally expensive than using a simple planning algorithm. Furthermore, they explore this approach only in an on-road driving case.

III. MODEL BASED REINFORCEMENT LEARNING

Reinforcement learning is an appropriate approach for solving Markov Decision Processes (MDPs). An MDP is formally defined as a tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}\}$, where \mathcal{S} is the state space, \mathcal{A} is the action space, \mathcal{T} is the transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow p(\mathcal{S})$, and \mathcal{R} is the reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$. In this formulation, we observe a state $s_t \in \mathcal{S}$ at time step t and the agent selects an action $a_t \in \mathcal{A}$. The environment returns the next state $s_{t+1} \in \mathcal{S}$ according to the transition function $\mathcal{T}(\cdot|s_t, a_t)$. The agent then receives a reward $r_t = \mathcal{R}(s_t, a_t, s_{t+1})$ representing the value of its current position in the environment.

The agent selects actions according to its policy $\pi : \mathcal{S} \rightarrow p(\mathcal{A})$. We define the state-action value function $Q^\pi(s, a)$ for a given policy π :

$$Q^\pi(s, a) = \mathbb{E}_{\pi, \tau} \left[\sum_{k=0}^K \gamma^k r_{t+k} | s_t = s, a_t = a \right] \quad (1)$$

where γ is the discount factor and K is the number of state transitions made. In its recursive form, $Q^\pi(s, a)$ is recognizable as the Bellman equation:

$$Q^\pi(s, a) = \mathbb{E}_{s' \sim \tau(\cdot|s, a)} + [\mathcal{R}(s, a, s') + \gamma \mathbb{E}_{a' \sim \pi(\cdot|s')} [Q^\pi(s', a')]] \quad (2)$$

In other words, $Q^\pi(s, a)$ is the expected cumulative return for π . An optimal policy π^* will maximize the state-action value function.

$$Q^{\pi^*}(s, a) = \operatorname{argmax}_\pi Q^\pi(s, a) \quad (3)$$

We know at least one such policy π^* must exist [8]. It is then the objective of the reinforcement learning algorithm to find an optimal policy.

In model-based reinforcement learning (MBRL) [9], rather than implicitly learning an environment's dynamics to compute an optimal policy π^* , an agent explicitly learns a dynamics model $(s_t, a_t) \rightarrow s_{t+1}$. This model is supervised by

experience (i.e. trajectories) collected from the environment. We can similarly train models to predict the action a_{t+1} that will be taken under a current policy or the value of the current state. In this way, we approximate the relevant aspects of the MDP simply by interacting with the environment. These models can then be used for traditional planning algorithms in environments with unknown dynamics.

IV. APPROACH

A. Dynamics Model

Large, outdoor environments are often too complex to explicitly model. Since off-road driving necessitates navigating such environments, we propose a MBRL approach to solve this problem. In our approach, the agent first learns a dynamics model of the environment by collecting experience within it. It subsequently uses this model in one of two traditional planning methods, described in detail below.

Specifically, we train a neural network model to predict relevant aspects of the environment from current state observations. The full model architecture is shown in Fig.1. The input state is made up of stacked observations from the environment. This state information is encoded as a latent vector and fed into three fully connected networks predicting the value of the current state, the policy at the current state, and the next state transition, respectively. Learning is supervised by state-action trajectories collected from the environment, and we train the model using back-propagation through time.

B. Latent Space Representation

When encoding the input state of this model, we must consider the expected observations. In our problem domain, autonomous off-road vehicles are equipped with a variety of sensors for collecting information about the environment. For example, LiDAR, IMU, image, and stereo sensors each provide different information relevant to decision making. To provide a general model that can be adapted to any number and modality of sensors, we propose using a latent vector representation for state encoding.

Latent dynamics models have been used to predict future states from high-dimensional observations [10]. These latent space models allow for simpler and faster computations than analogous image space models. We combine this method with the MBRL architecture described in Fig. 1 by using a latent space representation of multi-modal sensing data as input for the three networks. To construct the latent space, we use a convolutional neural network (CNN) model to extract latent information from sensor observations. This model is supervised by trajectories collected from the environment and trained according to a contrastive loss. By incorporating the accuracy and robustness that comes from high-dimensional sensor data, this low-dimensional latent space ultimately improves the accuracy of the dynamics model.

C. Planning

From this dynamics model, we can predict information about environment transitions necessary for planning. We

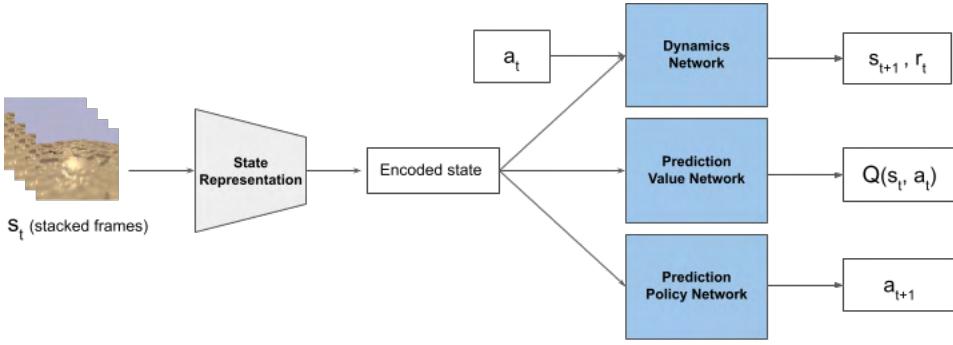


Fig. 1. MBRL Architecture. We predict the transition (i.e. next state, reward), value, and policy from the observation at time t . We can then roll out this model for use in traditional planning algorithms.

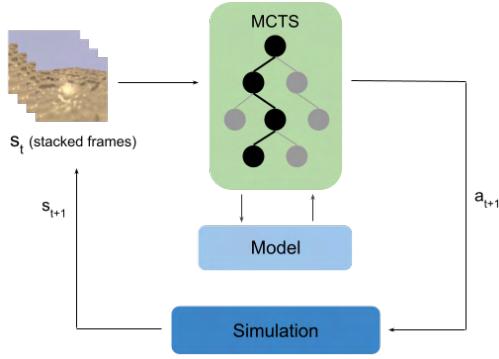


Fig. 2. MuZero Framework. The dynamics model is used to roll out state-action trajectories, and values are updated with back propagation.

present a MuZero framework using Monte Carlo Tree Search for incorporating our model into planning.

1) MuZero: The MuZero algorithm was proposed by [1] as a MBRL method for solving problems in complex domains without knowledge of their underlying dynamics. Similar to the architecture described in Fig. 1, in this algorithm a model is used to predict aspects of future state relevant for planning. Building on the success of lookahead search methods for planning, MuZero uses the model predictions in a Monte Carlo Tree Search (MCTS) to select optimal actions at each time step t .

We extend this framework to our problem domain using the network architecture described in Fig. 1. At each node in the tree search, we use our model to predict next state and reward. We roll out 50 iterations at each time step, and update the node values using back-propagation. The result of the MCTS is used to select an action and take a step in the environment. This updates the agent's experience, which in turn updates model training. A diagram of this process can be seen in Fig. 2. Because we are modeling only the aspects of state relevant to planning, this framework allows for efficient planning even in a complex and unknown environment. We evaluate preliminary results for this approach in Section V.

V. EXPERIMENTS

A. Lunar Lander

We test our MuZero framework on the OpenAI Gym environment ‘LunarLander-v2’ [11]. In this game, an aerial vehicle must properly land in a specified landing zone. While this scenario diverges from our ultimate off-road driving application, it is a toy example that allows us to model control parameters for a robotic vehicle. In this way, it provides a baseline for discussion. We further note in Section V our plans for adapting this method to the off-road driving application.

In this environment, an agent’s state is made up of the vehicle’s position, orientation, and velocity. The actions available to the agent are to fire the left engine, fire the right engine, fire the main engine, or do nothing. The reward is defined based on the orientation and position of the landed vehicle. Specifically, we set

$$r_t = \begin{cases} +100 & \text{if landed} \\ -100 & \text{if crashed} \\ -100 * |\text{velocity}_t| \\ -0.03 * \text{orientation}_t & \text{else} \\ -0.3 * \text{power}_t \end{cases}$$

where orientation_t and velocity_t represent the respective aspects of state at time t and power_t is based on the action taken at time t .

Following [12], we use fully connected networks with a hidden layer of size 64 to model each of dynamics, value, and policy. We use Adam optimization [13], a sigmoid activation function, and a log softmax loss. We define a constant learning rate $\lambda = 0.005$. We train for 50,000 steps with a batch size of 64. Training results are shown in Fig.3.

A positive final reward indicates the vehicle landed within the specified ‘landing zone.’ A perfect run corresponds to a reward of 200. Notice in Fig.3 that training converges to a positive reward, but never reaches the optimal value. We can see in Fig. 4 that, in testing, this model does achieve the goal of landing the vehicle in the desired zone. This result suggests that this method is valid to consider for the off-road driving problem. However, the vehicle does not take the most direct path, meaning its policy remains sub-optimal. We

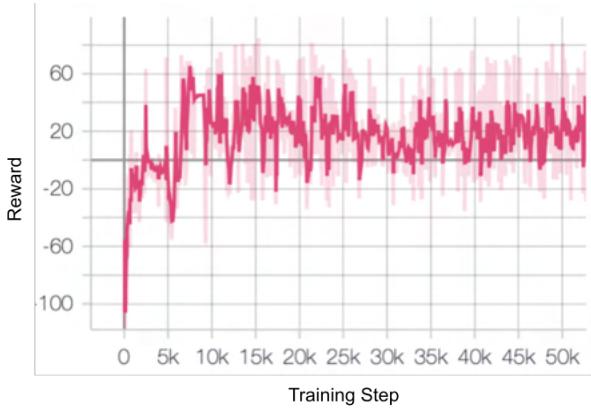


Fig. 3. Training performance for LunarLander-v2 simulation

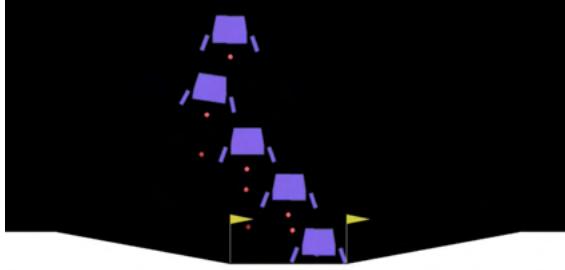


Fig. 4. Representative test trajectory for LunarLander-v2 environment. Each purple vehicle represents the agent’s position at a distinct point in time. Notice that the agent successfully lands in the landing zone, but does not take the most direct path.

postulate that the control problem may be better solved with a simpler method. For this reason, we consider a modified MuZero in the next section.

B. Robot Simulation

We additionally test the MuZero framework in a simulation environment for off-road driving. This simulation is run through a pybullet engine [14]. We generate height and friction maps to build simulated terrain. The goal in this environment is for the vehicle to reach a predefined target location.

Due to the inefficiencies discussed in the previous section, we test a modified MuZero approach in this environment. In this setup, we use only the predictions from the Dynamics Network. Rather than using model predictions for the value of the state, we calculate it directly from the reward function. We note that because this task is straightforward, it is appropriate to assume that we have full access to the reward function.

The agent’s state is made up of position, orientation, and the respective derivatives. We also extract the local height map and friction map for use in input state. Our framework is generalizable to include any other sensor information. The actions available to the agent are steer and throttle [steer, throttle] $\in (-1, 1) \times (-1, 1)$. Because the goal is to reach

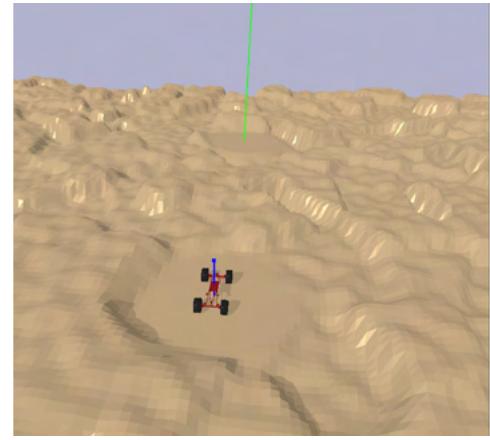


Fig. 5. Off-road driving simulation. In this task, the robot must reach the target indicated by the green laser.

a target position, the reward should encourage the agent to move towards the target. As such, we define the reward as the negative l_2 distance from the target.

Note that this study is still ongoing, and we plan to collect full experimental results as part of future work.

VI. FUTURE WORK

The results presented in this paper are still preliminary and have several limitations. We plan to implement the full MuZero approach in the pybullet simulation environment. As discussed in the previous section, in the ‘LunarLander-v2’ game the MuZero framework reaches satisfactory but not optimal reward. The highly complex model also requires 16 GiB of memory for training and currently takes 6 hours to reach convergence. Before extending this to the pybullet simulation, these inefficiencies must be solved.

In future work, we plan to evaluate different combinations of sensing modalities for improving state estimation. Furthermore, we want to incorporate considerations of uncertainty into the model. Finally, we plan to extend this setup for testing on a physical all terrain vehicle (ATV). The ATV will have the same task of reaching a target position over varying terrain, and will be equipped with LiDAR, GPS, IMU, Stereo, and Racepak systems. Physical experiments will allow us to evaluate the approach in a more complex environment with greater uncertainty and to explore a greater range of sensing modalities.

VII. CONCLUSION

Autonomous off-road driving is a difficult task because the size and complexity of off-road environments makes their dynamics difficult to explicitly model. We have proposed a model-based reinforcement learning approach to learn a model of the environment by interacting with it. Traditional planning, such as optimization or MCTS, can be done on this model to select the best actions. We demonstrated preliminary results in two simulation environments, each suggesting promise for further study. Ultimately, this work can be used in applications such as search and rescue and

ground exploration. It is critical that algorithms for such applications be both safe and efficient, further motivating the development of our approach.

ACKNOWLEDGMENT

Thanks to the Robotics Institute Summer Scholars (RISS) program at CMU, the RISS sponsors, and my mentors for their support and help making this research possible.

REFERENCES

- [1] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. P. Lillicrap, and D. Silver, "Mastering atari, go, chess and shogi by planning with a learned model," *CoRR*, vol. abs/1911.08265, 2019. [Online]. Available: <http://arxiv.org/abs/1911.08265>
- [2] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, 2011, pp. 163–168.
- [3] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21918>
- [4] M. Jaritz, R. de Charette, M. Toromanoff, E. Perot, and F. Nashashibi, "End-to-end race driving with deep reinforcement learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2070–2075.
- [5] M. Bajracharya, A. Howard, L. H. Matthies, B. Tang, and M. Turmon, "Autonomous off-road navigation with end-to-end learning for the lagr program," *Journal of Field Robotics*, vol. 26, no. 1, pp. 3–25, 2009. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20269>
- [6] N. Rhinehart, R. McAllister, and S. Levine, "Deep imitative models for flexible inference, planning, and control," *CoRR*, vol. abs/1810.06544, 2018. [Online]. Available: <http://arxiv.org/abs/1810.06544>
- [7] J. Wu, Z. Huang, and C. Lv, "Uncertainty-aware model-based reinforcement learning with application to autonomous driving," *CoRR*, vol. abs/2106.12194, 2021. [Online]. Available: <https://arxiv.org/abs/2106.12194>
- [8] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. The MIT Press, 2018. [Online]. Available: <http://incompleteideas.net/book/the-book-2nd.html>
- [9] T. M. Moerland, J. Broekens, and C. M. Jonker, "Model-based reinforcement learning: A survey," *CoRR*, vol. abs/2006.16712, 2020. [Online]. Available: <https://arxiv.org/abs/2006.16712>
- [10] D. Hafner, T. P. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *CoRR*, vol. abs/1912.01603, 2019. [Online]. Available: <http://arxiv.org/abs/1912.01603>
- [11] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.
- [12] A. H. Werner Duvaud, "MuZero general: Open reimplementation of muzero," <https://github.com/werner-duvaud/muzero-general>, 2019.
- [13] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.
- [14] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," <http://pybullet.org>, 2016–2021.
- [15] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/0005109889900022>
- [16] D. Maturana, P.-W. Chou, M. Uenoyama, and S. Scherer, "Real-time semantic mapping for autonomous off-road navigation," in *Field and Service Robotics*, M. Hutter and R. Siegwart, Eds. Cham: Springer International Publishing, 2018, pp. 335–350.
- [17] M. Himmelsbach, T. Luettel, F. Hecker, F. Hundelshausen, and H.-J. Wuensche, "Autonomous off-road navigation for mucar-3 - improving the tentacles approach: Integral structures for sensing and motion," *KI*, vol. 25, pp. 145–149, 05 2011.

ROCCER Evaluation Within Federated Classifier Selection for Improved Identification of Center Collaboration Opportunities

Willa Potosnak^{1,2}, Sebastian Caldas, M.Sc.², James K. Miller, Ph.D.²
and Artur Dubrawski, Ph.D.²

Abstract—Data heterogeneity among hospital centers due to varying quality of care highlights the need for collaboration among centers. An explicit and translatable understanding of center expertise is important to promote collaboration among centers for best practice adaptation and risk model development. Decision lists are interpretable models and thus, have considerable value for this purpose. The Federated Classifier Selection (FRCLS) algorithm leverages decision lists for use in providing insight into beneficial collaborations among centers. To generate a decision list for this purpose, FRCLS performs rule selection using a heuristic that maximizes the lower bound on the mean (MLBM) of a target variable. ROCCER is an alternative rule selection algorithm that uses receiver operating characteristic (ROC) analysis for the purpose of maximizing area under the ROC curve (AUC). Optimizing for AUC can result in more robust models for data with smaller sample sizes. This research contributes a novel application of ROCCER within FRCLS to assess if rule selection using ROC analysis generates decision lists with improved insight into potentially beneficial collaborations among centers. The results from this research show that rule selection using ROC analysis can potentially improve decision list rule utility for certain center collaborations.

I. INTRODUCTION

Data is inherently heterogeneous between hospital centers due to varying patient population characteristics and hospital practices [1], [2]. As patient population characteristics and practices vary between hospitals, centers may specialize to certain populations or sub-populations and so possess expertise that can be shared to benefit other centers [2]. This expertise may be easily distinguished, such as for certain disease institutes, and thus is already understood and shared between centers. Expertise could also be a result of the center's adaptation to their local population and its needs. Expertise of this type is not easily identifiable, which prevents this useful information from being leveraged by other centers. Identifying expertise of this type could lead to improved quality of care and better outcomes for patients through collaboration among centers [1].

An explicit and translatable understanding of center ex-

¹Willa Potosnak is a student in her 4th year in the Biomedical Engineering Department at Duquesne University, Pittsburgh, PA, USA potosnakw@duq.edu

²Sebastian Caldas, M.Sc., James K. Miller, Ph.D. and Artur Dubrawski, Ph.D. are with the Auton Lab, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA scaldas@cmu.edu, mille856@andrew.cmu.edu, awd@cs.cmu.edu

pertise is important to promote collaboration among centers for best practice adaptation and risk model development. However, understanding which collaborations are beneficial is also necessary as not all external expertise may be useful for a particular center. Machine learning (ML) can be used to determine center expertise that could result in beneficial collaborations. It is crucial that ML models applied for this purpose be both reliable and interpretable if they are to gain clinician trust and ultimately promote collaboration among centers when it can prove beneficial. Decision lists are interpretable models and thus, have considerable value for this purpose.

This research contributes a novel application of ROCCER within Federated Classifier Selection (FRCLS) to assess if rule selection using receiver operating characteristic (ROC) analysis generates decision lists with improved insight into potentially beneficial collaborations among centers.

II. CONTEXT

A. Decision List

Decision lists are ML models that comprise an ordered list of rules. The appeal of decision lists are the explicit rules that make them interpretable for regression or classification tasks and their ability to generalize decision trees in their simple list-like structure shown in figure 1 [3]. Decision list construction is composed of two tasks, rule generation and rule selection.

Rule generation is the process of producing rules for potential inclusion in the decision list. The rule generation algorithm of particular interest for this research is an implementation of the RADSEARCH algorithm [4]. The implementation can function as an association rule learning algorithm to generate rules for a specified target variable and is equipped to handle both binary and real-valued targets. To generate rules, it requires the user set certain constraints in the form of hyper-parameters to filter rules. As decision list generation is an NP-complete problem [5], [6], the use of heuristics is generally applied to guide rule selection or pruning. Different rule heuristics have been applied in research for this purpose such as accuracy [7], entropy [8], description length [9], [10], and area under ROC curve

(AUC) [6], [11].

ROCCER is a rule selection algorithm that uses ROC analysis for the purpose of maximizing AUC of the decision list model [11]. ROCCER was chosen as an rule selection method to evaluate within FRCLS for the following reasons:

- 1) Optimizing for AUC can result in more robust models for data with smaller sample sizes [12]. This has applications for research with smaller hospital centers as these especially may derive benefit from center collaboration.
- 2) In using ROC analysis, ROCCER compares potential rules to a default rule with random performance to ensure the inclusion of a rule in the decision list offers improvement over random decision making.
- 3) ROCCER allows for disjoint rule sets to be selected in place of a single decision list rule, which could allow for clinicians to customize the model by choosing more relevant rules for their specific healthcare purposes.

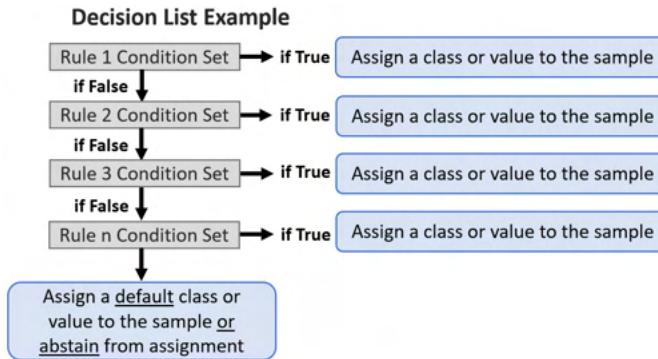


Fig. 1: A decision list is an ML model that comprises a list of rules. It can be used for classification or regression tasks where rules are tried consecutively in the list until a rule or no rule describes the queried sample.

B. Federated Classifier Selection

The FRCLS algorithm is based on the idea that center practices or models may be more robust for particular sub-populations [2]. This algorithm generates a decision list that dynamically guides classification model selection for patients as shown in figure 2. In addition to model selection, the rules generated by this algorithm have particular application for providing insight into when a classification model developed using data from the external center is more competent for patients within a particular region of the feature space than a classification model developed using data from the local center [2]. This insight can be used to determine center expertise as well as potentially beneficial center collaborations.

FRCLS generates a decision list for the target variable $\ln(pE)$ as outlined in [2] where pE can be described as a ratio of local model to external model average cross-

entropy losses computed for k-nearest neighbors of the selected sample. After FRCLS generates a decision list, it iterates through the list and computes a binomial test p-value from the cumulative number of successfully flipped samples. (Successfully flipped samples have incorrectly predicted outcomes by the local model but are described by a decision list rule and directed to the external model which predicts a correct outcome. Failure flipped samples are those with the reverse predicted outcomes in the previous scenario.) After FRCLS iterates through the decision list and computes p-values for each rule, it truncates the decision list after the rule with the smallest p-value result. If the the number of successfully flipped samples is statistically significant ($p\text{-value} < 0.05$), then the decision list is recommended to guide model selection.

Reliable rule selection for decision list generation within FRCLS is important to better determine potentially beneficial collaborations among centers and increase user trust in the recommended collaboration. FRCLS incorporates an implementation of the RADSEARCH algorithm [4] for rule generation. To generate a decision list, FRCLS performs rule selection using a heuristic that maximizes the lower bound on the mean of $\ln(pE)$ [2]. This paper will refer to this rule selection method as MLBM.

III. DATA

Two datasets were used to evaluate the application of each rule selection method within FRCLS. The first dataset was processed using data collected from the University of Pittsburgh Medical Center (UPMC) under a research project titled, 'Machine Learning of Physiological Variables to Predict, Diagnose and Test Cardio-respiratory Instability,' or MLADI. The second dataset was used for a case study on indwelling arterial catheters and procured from the MIMIC-II database through PhysioNet [13]–[15].

A. MLADI

This paper refers to data from the MLADI project as 'MLADI'. The MLADI data was used for a classification task to predict the binary outcome of a first hypotension episode one hour in advance. For this research, a hypotension episode was defined as at least 2 hypotension events no more than 1 hour apart where a hypotension event was defined as systolic blood pressure less than 90 mmHg and mean arterial pressure less than 65 mmHg. The processed data consists of 1,563 patients and 25 features of both patient demographics and physiological measurements. The 15 minutes of data prior to one hour before the first hypotension episode was used to generate features for patients with this outcome. For patients without a hypotension outcome, the median time after the first two hours from admission until the first

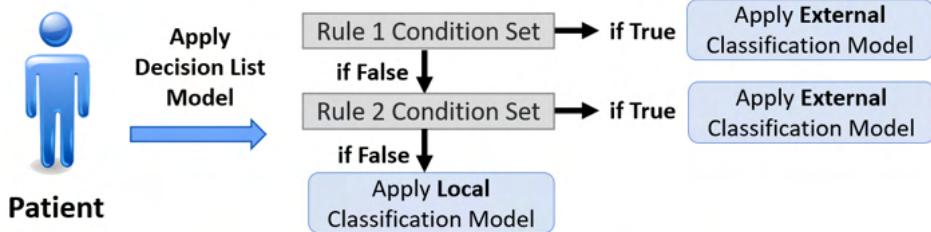


Fig. 2: FRCLS generates a decision list that can be used to guide model selection for patients. For patients described by a rule in the decision list, the classification model developed using data from the external center is recommended. If no rule in the list describes the patient, then the default model developed using data from the local center is recommended. In addition to guiding model selection, the rules provide insight for when the external model is more competent for patients within a particular region of the feature space than the local model.

hypotension episode was used to determine the 15 minutes of data used for feature generation. Two intensive care units (ICU) of medical (MICU) and surgical (SICU) are present and used to partition the data into artificial data silos for application with FRCLS. The MICU silo consists of 590 patients with 224 labeled ‘positive’ for hypotension outcome. The SICU silo consists of 973 patients with 336 labeled ‘positive’ for this outcome.

B. MIMIC-II

Data was extracted from the MIMIC-II database [14], [15] and recreated for use in a case study in [13]. This paper refers to the data as ‘MIMIC-II’. The classification task of this data is to predict the binary outcome of death within 28 days of admission as this was the main outcome of interest in [13]. It consists of 1,776 patients and 31 features of both demographics and physiological measurements. Two ICU units, MICU and SICU, are present and used to partition the data into artificial data silos for application with FRCLS. The MICU silo consists of 732 patients with 127 labeled ‘positive’ for the outcome of death within 28 days of admission. The SICU silo consists of 982 patients with 153 labeled ‘positive’ for this outcome.

IV. METHOD

A. Data Partitioning

The ICU stay descriptor variable was used to partition patients into two artificial data silos of ‘MICU’ and ‘SICU’ where each silo’s data were considered separate centers. After the centers were generated, each center’s data was split into training, validation, and test sets for the respective proportions of the data, 10%, 45%, 45%. The training data was proportioned at 10% to allocate more samples for decision list generation and evaluation. The training set was used for model development where random forest algorithms were

trained on each center’s data using 5-fold cross-validation for hyper-parameter selection. The validation set was used for decision list generation through FRCLS and the test set was used for a final evaluation of the decision list on unseen data.

B. Rule Selection Algorithm

The ROCCER algorithm was re-created from the methods described in [11]. An implementation of the RADSEARCH algorithm proposed in [4] was used to generate rules. MLMB and ROCCER were evaluated within FRCLS to assess if rule selection using ROC analysis generates decision lists with improved insight into potentially beneficial collaborations among centers.

C. Evaluation Metrics

Each center collaboration was evaluated. More specifically, FRCLS was evaluated for when the MICU data and model were considered ‘local’ and the SICU data and model were considered ‘external’ and vice versa as shown in figure 3. The decision list generated using each rule selection method within FRCLS was evaluated to determine if the number of patients with predictions flipped from incorrect to correct using the external classifier is statistically significant using a binomial test [2]. Significant results ($p\text{-value} < 0.05$) would indicate the utility of the decision list rules to discern patients who would benefit from the use of the external model rather than the local model. The following results for the validation and test set were obtained over a range of random forest model false positives rate (FPR) values:

- Binomial test p-value
- Number of samples described by the decision list rules and directed to the external classifier
- Local and external model accuracy for samples described by the decision list rules

Python 3.8 (Python Software Foundation, Beaverton, Oregon) and scikit-learn software [16] were used in this research.

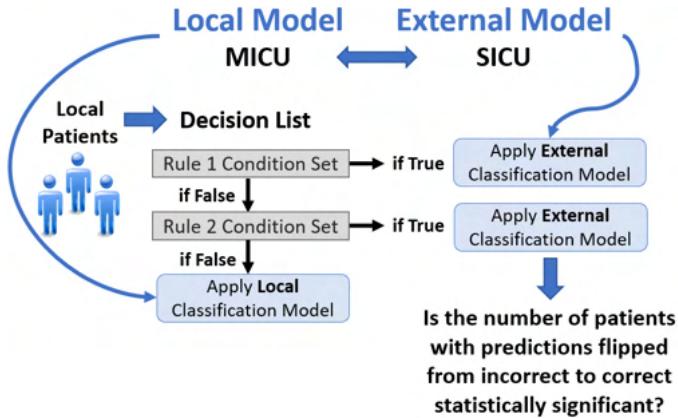


Fig. 3: The decision list generated using each rule selection method within FRCLS was evaluated to determine if the number of patients with predictions flipped from incorrect to correct using the external model is statistically significant using a binomial test. Significant results would indicate the utility of the decision list rules to discern patients who would benefit from use of the external model rather than the local model.

V. RESULTS

A. MLADI

Rule selection using MLBM generates shorter decision lists with higher rule support (the number of samples described by the rule) than the decision lists generated using ROCCER as shown in tables II and III of the appendix. The FRCLS test set results for MICU as the local center and SICU as the external center in figure 4 (a) show FPR regions with significant p-values for both rule selection methods. The FRCLS test set results for SICU as the local center and MICU as the external center in figure 4 (d) show a larger range of significant p-value results for MLBM at an approximate FPR range of 30% to 60% compared to the results for ROCCER. For patients described by decision list rules selected using either MLBM or ROCCER, the external model has a higher accuracy than the local model as shown in figure 4 (f).

B. MIMIC-II

Rule selection using MLBM generates shorter decision lists than the decision lists generated using ROCCER yet with comparable rule support as shown in tables IV and V of the appendix. The FRCLS test set results for MICU as the local center and SICU as the external center in figure 5 (a) show FPR regions with significant p-values for MLBM. The

FRCLS test set results for the SICU as the local center and MICU as the external center in figure 5 (d) show significant p-values for ROCCER following a FPR of 40%. For patients described by the decision list rules selected using ROCCER, the external model has a higher accuracy than the local model as well as a higher accuracy compared to the results for MLBM as shown in figure 5 (f)

VI. DISCUSSION

The FRCLS test set results for the decision list produced using ROCCER show FPR regions with statistically significant p-values. This is most notable for the MIMIC-II data center collaboration of SICU as the local center and MICU as the external center. This could indicate that ROCCER has utility in discerning patients who would benefit from use of the external model rather than the local model. In practice, this information can be used to infer that the MICU external center may have expertise that could benefit the SICU local center through collaboration. The FRCLS test set results for the decision list generated using MLBM also show FPR regions with statistically significant p-values. This is most notable for the MLADI data center collaboration of SICU as the local center and MICU as the external center. Interestingly, the rule selection methods show few overlapping FPR regions of statistical significance. This could indicate that the rule selection methods may individually perform better within FRCLS for particular data.

VII. CONCLUSION

Rule selection using ROC analysis can potentially improve rule utility over MLBM for certain center collaborations. Significant results at more model FPR values, particularly within 10%, would be needed to help confirm this conclusion and the application of ROCCER within FRCLS. Future work will include incorporating rule significance testing in ROCCER to ensure reliable rule inclusion in the decision list and performing additional evaluations on larger datasets. Research regarding best practice discovery and FRCLS analysis with artificial data silo partitions based on time period will also be explored.

VIII. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1659774. Thank you to Carnegie Mellon University Robotics Institute for supporting this research as well as Ms. Rachel Burcin and Dr. John Dolan for their work to help make the RISS program possible. A special thanks to Dr. Artur Dubrawski and Dr. Kyle Miller for their invaluable mentorship and guidance throughout this research.

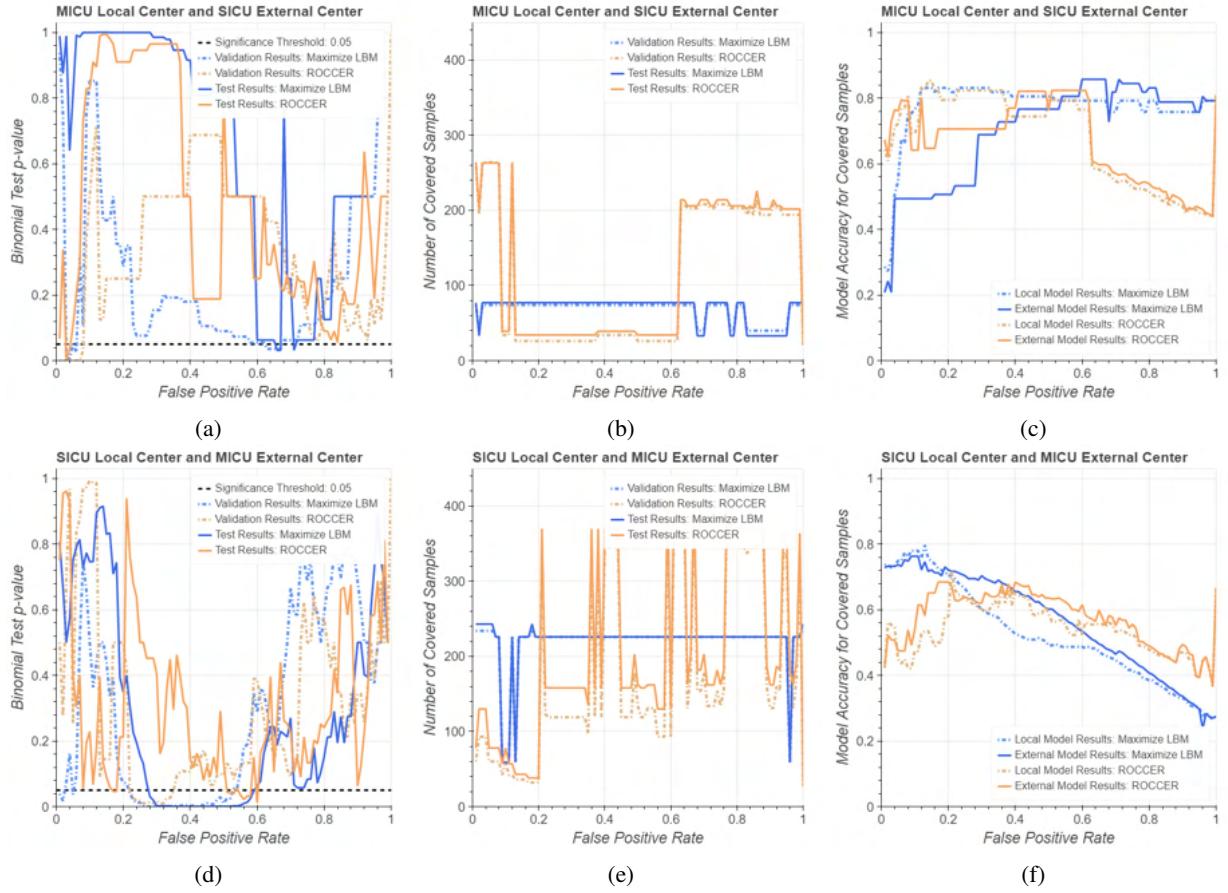


Fig. 4: FRCLS results for MLADI data computed for a range of random forest model FPR values are shown for decision lists generated using MLBM (blue) and ROCCER (orange). The top row of plots corresponds to MICU as the local center and SICU as the external center. The bottom row of plots corresponds to SICU as the local center and MICU as the external center. Plots (a) and (d) show the binomial test p-values for the validation (dashed line) and test (solid line) sets. Plots (b) and (e) show the number of samples described by the decision list for the validation (dashed line) and test (solid line) sets. Plots (c) and (f) show the local (dashed line) and external (solid line) random forest model accuracy for samples described by the decision list for the test set. Plot (d) shows a larger range of significant p-value results for MLBM at an approximate FPR range of 30% to 60% compared to the results for ROCCER. For patients described by decision list rules selected using either MLBM or ROCCER, the external model has a higher accuracy than the local model as shown in (f).

REFERENCES

- [1] M. Ali, R. Salehnejad, and M. Mansur, “Hospital heterogeneity: what drives the quality of health care,” *European Journal of Health Economics*, no. 3, pp. 385–408, Apr. 2018.
- [2] S. Caldas and A. Dubrawski, “Understanding clinical collaborations through federated classifier selection,” in *Proc. of Machine Learning Research*, 2021.
- [3] R. L. Rivest, “Learning decision lists,” *Machine Learning*, pp. 229–246, 1987.
- [4] A. Moore and J. Schneider, “Real-valued all-dimensions search: Low-overhead rapid searching over subsets of attributes,” in *Proc. of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, 2002, pp. 360–369.
- [5] T. Hancock, T. Jiang, M. Li, and J. Tromp, “Lower bounds on learning decision lists and trees,” *Information and Computation*, no. 2, pp. 114–122, 1996.
- [6] T. Fawcett, “Prie: A system for generating rulelists to maximize roc performance,” *Data Mining and Knowledge Discovery*, no. 2, pp. 207–224, Oct. 2008.
- [7] R. Segal and O. Etzioni, in *Learning Decision Lists Using Homogeneous Rules*, 1994, pp. 619–625.
- [8] P. Clark and T. Niblett, “The cn2 induction algorithm,” *Machine Learning*, vol. 3, pp. 261–283, 1989.
- [9] W. W. Cohen, “Fast effective rule induction,” in *Proc. of the Twelfth International Conference on Machine Learning*, 1995, pp. 115–123.
- [10] W. W. Cohen and Y. Singer, “A simple, fast, and effective rule learner,” in *AAAI-99*, 1999, pp. 335–342.
- [11] R. C. Prati and P. A. Flach, “Roccer: an algorithm for rule learning based on roc analysis,” in *Proc. of the 19th International Joint Conference on Artificial Intelligence*, 2005, pp. 823–828.
- [12] T. Fawcett, “Using rule sets to maximize roc performance,” in *Proc. 2001 IEEE International Conference on Data Mining*, 2001, pp. 131–138.
- [13] J. Raffa, M. Ghassemi, T. Naumann, M. Feng, and D. Hsu, *Data Analysis*, 2016.
- [14] J. Raffa, “Clinical data from the mimic-ii database for a case study on indwelling arterial catheters (version 1.0),” 2016.
- [15] A. Goldberger, L. Amaral, L. Glass, J. Hausdorff, P. C. Ivanov, R. Mark, and et al, “Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals,” *Circulation*, p. e215–e220, 2000.
- [16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and

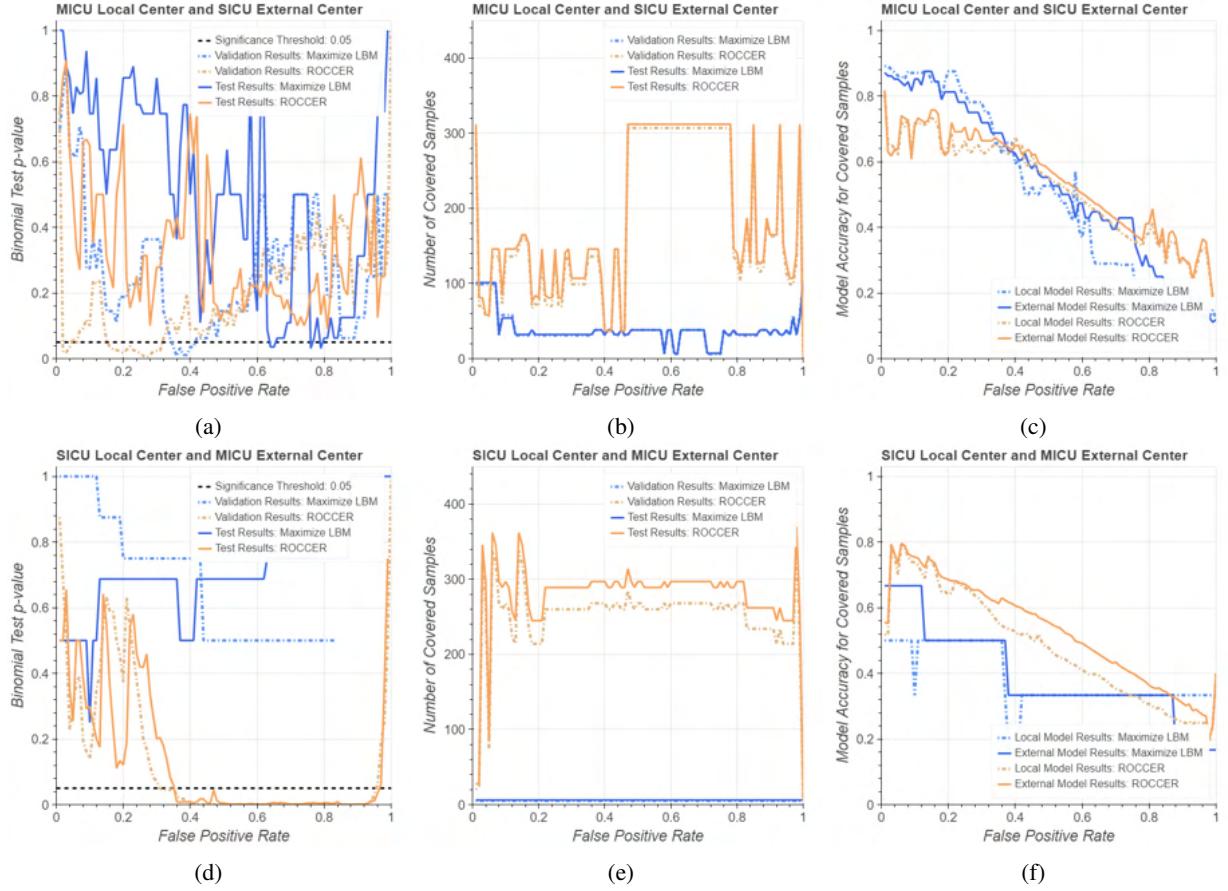


Fig. 5: FRCLS results for MIMIC-II data computed for a range of random forest model FPR values are shown for decision lists generated using MLBM (blue) and ROCCER (orange). The top row of plots corresponds to MICU as the local center and SICU as the external center. The bottom row of plots corresponds to SICU as the local center and MICU as the external center. Plots (a) and (d) show the binomial test p-values for the validation (dashed line) and test (solid line) sets. Plots (b) and (e) show the number of samples described by the decision list for the validation (dashed line) and test (solid line) sets. Plots (c) and (f) show the local (dashed line) and external (solid line) random forest model accuracy for samples described by the decision list for the test set. Plot (d) shows significant p-values for ROCCER following a FPR of 40%. For patients described by the decision list rules selected using ROCCER, the external model has a higher accuracy than the local model as well as a higher accuracy compared to the results for MLBM as shown in (f).

- E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [17] N. Rieke, J. Hancox, W. Li, and et al, “The future of digital health with federated learning,” *npj Digital Medicine*, Sept. 2020.
- [18] S. Caldas, J. Chen, and A. Dubrawski, “Using machine learning to support transfer of best practices in healthcare,” 2021.

IX. APPENDIX

Code for the re-created ROCCER algorithm used in this research is available at the following GitHub repository link: https://github.com/PotosnakW/ROCCER_Implementation.git

Abbreviation	Feature
MAP	Mean arterial pressure
RR	Respiratory rate
SBP	Systolic blood pressure
DBP	Diastolic blood pressure
HR	Heart rate
BUN	Blood urea nitrogen

TABLE I: Abbreviations for decision list features

MLBM			
Rule Number	Rule Features	Rule Conditions	Rule Support (Val,Test)
1	MAP median RR median	≤ 80 mmHg > 20 br/m	40, 33
2	SBP median HR median	≤ 100 mmHg ≤ 80 bpm	34, 44

ROCCER			
Rule Number	Rule Features	Rule Conditions	Rule Support (Val,Test)
1	SBP mean DBP median	> 80 and ≤ 160 mmHg ≤ 50 mmHg	15, 18
2	Ethnicity RR median	Hispanic or Latino ≤ 14 br/m	4, 3
3	Race RR median	White > 28 br/m	7, 13
...

TABLE II: Decision lists generated using MLBM and ROCCER for the **MLADI** MICU local center to guide the use of the SICU external model over the MICU local model. Rule selection using MLBM results in a shorter decision list with larger rule support (the number of samples described by the decision list rule) than the decision list produced using ROCCER.

MLBM			
Rule Number	Rule Features	Rule Conditions	Rule Support (Val,Test)
1	SBP median HR median	> 100 mmHg > 80 bpm	58, 59
2	SBP median DBP median	> 100 mmHg ≤ 80 mmHg	168, 167
3	Race HR SD	Declined response ≤ 4 bpm	8, 17

ROCCER			
Rule Number	Rule Features	Rule Conditions	Rule Support (Val,Test)
1	SBP median DBP median	> 80 and ≤ 160 mmHg ≤ 50 mmHg	27, 27
2	DBP median MAP median	> 100 mmHg > 120 mmHg	5, 11
3	Age Race	> 75 years African American	4, 5
...

TABLE III: Decision lists generated using MLBM and ROCCER for the **MLADI** SICU local center to guide the use of the MICU external model over the SICU local model. Rule selection using MLBM results in a shorter decision list with larger rule support than the decision list produced using ROCCER.

MLBM			
Rule Number	Rule Features	Rule Conditions	Rule Support (Val,Test)
1	ICU admit hour MAP	> 12 > 120 mmHg	6, 7
2	SAPS I score Bicarbonate	≤ 14 > 30 mEq/L	25, 25
3	Weight ICU admit day	> 140 kg Not Sunday	7, 6
...

ROCCER			
Rule Number	Rule Features	Rule Conditions	Rule Support (Val,Test)
1	Age ICU admit hour	> 90 years > 16	6, 4
2	SOFA score MAP	> 10 > 80 and ≤ 160 mmHg	4, 6
3	SAPS I score Bicarbonate	> 20 ≤ 20 mEq/L	8, 4
...

TABLE IV: Decision lists generated using MLBM and ROCCER for the **MIMIC-II** MICU local center to guide the use of the SICU external model over the MICU local model. Rule selection using MLBM results in a shorter decision list with comparable rule support to the decision list produced using ROCCER.

MLBM			
Rule Number	Rule Features	Rule Conditions	Rule Support (Val,Test)
1	BUN	> 50 mg/dL	5, 6

ROCCER			
Rule Number	Rule Features	Rule Conditions	Rule Support (Val,Test)
1	SAPS I score PaO ₂	> 20 > 400 mmHg	7, 5
2	SAPS I score PaO ₂	> 20 > 200 and ≤ 400 mmHg	8, 10
3	SAPS I score Bicarbonate	> 20 > 12 and ≤ 24 mEq/L	1, 6
...

TABLE V: Decision lists generated using MLBM and ROCCER for the **MIMIC-II** SICU local center to guide the use of the MICU external model over the SICU local model. Rule selection using MLBM results in a shorter decision list with comparable rule support to the decision list produced using ROCCER.

Learning with Noisy Camera Extrinsic for Robust and Real-Time Omnidirectional Depth Prediction

Conner Pulling¹, Yaoyu Hu², and Sebastian Scherer³

Abstract—Real-time omnidirectional depth prediction is vital for aerial robotics to accurately perceive their surroundings. Among methods that use multiple fisheye camera images as inputs, recent work using deep learning has shown promising performance, but is not real-time. While recent non-learning methods have achieved real-time performance, all previous work fundamentally assumes constant camera positions with no calibration errors. This is not always a practical assumption, especially for low-cost drones and when using standard multi-camera calibration tools. This work presents a novel deep learning model capable of real-time, robust omnidirectional depth prediction using multiple fisheye images while maintaining low GPU memory consumption. To achieve robustness against calibration errors, this work utilizes a novel synthetic dataset that uses noisy camera extrinsics. With a low memory footprint, fast runtime, and robustness against calibration errors, this model architecture takes a step towards enabling omnidirectional depth prediction capabilities for low-cost drone autonomy applications.

Index Terms—Datasets for Robotic Vision, Deep Learning for Visual Perception, Omnidirectional Vision, Aerial Systems, Perception and Autonomy, Field Robots

I. INTRODUCTION

Efficient and reliable scene understanding is a principle technology that drives advances in unmanned aerial vehicle (UAV) autonomy. Applications such as large infrastructure inspection [1] and agriculture [2] rely on dense depth predictions to provide the agent with 3D scene understanding. UAVs used for these applications need a constant flow of omnidirectional depth predictions to react to all possible obstacles in real-time. Recent work has proposed deep learning approaches that use multiple fisheye images to perform omnidirectional depth estimation with success, but fail to reach real-time inference times [3] [4]. Similar to recent non-learning methods, these deep-learning methods assume one set of camera positions as the ground truth and use this singular extrinsic configuration throughout training and evaluation. This assumption is not realistic as calibration errors are all too common in practical applications of drones and multi-camera configurations are often difficult to calibrate.

To address robustness and real-time concerns, this work presents a novel deep learning architecture that operates in

¹Conner Pulling is a Robotics Institute Summer Scholar at Carnegie Mellon University and an undergraduate Mechanical Engineering senior at Place. cwilliampulling@vt.edu

²Yaoyu Hu is a Postdoctoral Researcher with AirLabs at the Robotics Institute in Carnegie Mellon University. yaoyuh@andrew.cmu.edu

³Sebastian Scherer is an Associate Professor at the Robotics Institute in Carnegie Mellon University and the Principal Investigator at AirLabs. basti@andrew.cmu.edu

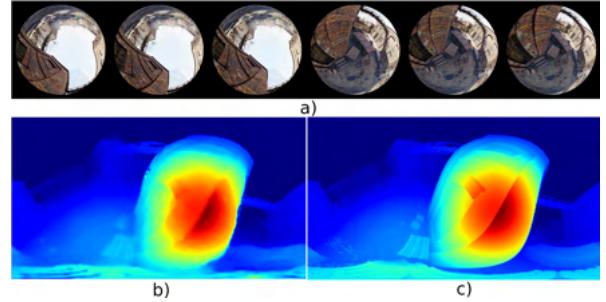


Fig. 1. Overview. a) 686-by-686 Fisheye Images that serve as input to the deep learning model. b) The predicted depth panorama by our deep learning model. c) The ground truth panorama generated by AirSim. Warmer colors indicate depths closer to the rig origin frame.

real-time by reducing the resolution of the final feature maps and drastically reducing the depth of the cost volume after spherical sweeping. This work shows that sweeping with lower sphere count drastically speeds up inference time while actually slightly improving accuracy. Additionally, the model was trained on a novel synthetic dataset that adds noise to the camera extrinsics while still achieving good performance metrics. This work shows that noisy camera extrinsics in the training data makes the model more robust against calibration errors.

In summary, this work presents the following contributions:

- This work presents a deep learning model that can inference at real-time while achieving comparable quality in the depth panorama outputs.
- This work proposes a new synthetic dataset of over 3k samples collected by AirSim that adds noise to the camera extrinsics.

II. RELATED WORK

A. Conventional Binocular Stereo Vision

Typically, most conventional binocular stereo methods that produce dense predictions take two rectified images as input where one image is called the reference. The goal of conventional methods is to find the disparity between the reference and pair image. With a known relative distance between the two cameras, called the baseline, disparity is defined as the distance in pixels between correspondences in each image. For dense depth prediction, disparity must be found for every pixel in the image. One of the most popular non-learning methods is Semi-Global Matching (SGM), which generates disparities by minimizing an energy function [5]. The Pyramid Stereo Matching Network (PSMN) is popular

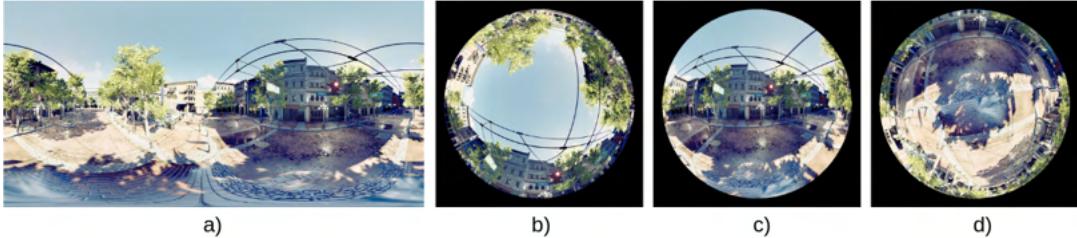


Fig. 2. **Comparison of fisheye distortion between different camera orientations.** a) Source Panorama Image, generated by AirSim. b) Fisheye Image generated with vertical orientation, pointing upwards. c) Fisheye Image generated with outward orientation, facing forward. d) Fisheye Image generated with vertical orientation, pointing downwards. Black border indicates regions outside of the fisheye camera's FOV.

learning-based architecture that focuses on evaluating the two input images at different spatial scales through spatial pyramid pooling to exploit global context information [6]. Another popular method for high-resolution depth predictions evaluates inputs at different resolutions for multi-stage output that can prioritize depth maps for the scene closer to the agent [7]. Hybrid methods have begun to use deep learning to guide non-learning methods, providing improved performance over either learning or non-learning methods [8].

The issue arises that conventional binocular stereo vision systems using planar lens cameras have narrow FOVs and cannot effectively capture visual information in all directions. Therefore, a multi-camera system is needed and visual information from multiple views must be fused into one depth map panorama.

B. Omnidirectional Multiview Stereo Vision

In hopes of better aggregating information from multiple stereo pairs, some works have turned to using learning-based approaches, inspired by the performance gains that deep learning has achieved for binocular stereo vision applications [6] [7]. SweepNet proposed warping the input fisheye images into the panorama space and used pairwise matching to compute the cost volume [9]. OmniMVS proposed warping the feature maps after strided convolutions to reduce memory and resource consumption before using an encoder-decoder architecture to regularize the cost volume [3]. CrownConv proposed projecting the fisheye image onto an icosahedron and using icosahedron-based spherical sweeping to be more computationally efficient [4]. However, none of the current learning-based omnidirectional stereo vision methods using multiple fisheye images have achieved real-time performance. Recent work has developed a real-time non-learning method to perform omnidirectional depth prediction, but does so by selectively using only the best stereo pair per pixel in a reference image and computes the cost volume only by using pixel intensities [10].

The current literature typically assumes a constant camera configuration with no calibration errors. While tools exist for multi-camera calibration, the process is often difficult, and prone to human error and approximation. Especially for low-cost drone applications, calibrations errors must be considered and mitigated.

C. Multi-Camera Calibration

Camera models have been developed to correctly account for distortion when rectifying or warping fisheye images. These models have a set of parameters such as focal length and image center that must be calibrated using tools like Kalibr [11]. The Pinhole Camera Model is the standard camera model to relate 3D physical coordinates to image coordinates and visa versa, often used with planar cameras. However, the pinhole camera model drastically becomes less accurate for FOVs greater than 120° [12]. The highest performing omnidirectional depth estimation methods use the Double Sphere camera model [10] [3].

D. Synthetic Multiview Fisheye Datasets

The most notable omnidirectional depth prediction datasets are OmniThings and OmniHouse [3] where each sample consists of four fisheye camera images that were generated in Blender. However, the dataset assumes fixed camera extrinsics and intrinsics, so the dataset is only useful for a specific camera configuration. This work provides the base panorama images as part of the presented dataset so that other works can warp the source into fisheye images specific to their camera orientations. Additionally, while the general location of each camera is fixed, noise was added to the camera location so a trained model would have to be robust against changes in camera extrinsics.

III. MULTI-RESOLUTION 360° STEREO

A. Reference Frames and Preprocessing

The predicted depth by the deep learning model uses the rig coordinate frame as the origin while limiting the minimum depth distance at 0.5 meters so the distance range can be discretized. Before preprocessing, RGB panoramas were collected with depth annotations using Unreal Engine 4 and a drone flight simulator called AirSim. AirSim uses a North, East, and Down (NED) coordinate frame as shown in Fig. 3 and the drone world position labels in the dataset follow these conventions. To use the rig coordinate frame as the origin, each camera has to be transformed to find the relative translation from the rig. As seen in Fig. 3, $T_{wr} | R_{wr}$ is the world to rig extrinsic camera matrix, T_{wb} is the world to camera translation vector, T^{rb} is the relative translation of the camera with the rig frame as the origin. Therefore, the relative translation is found with the equation,

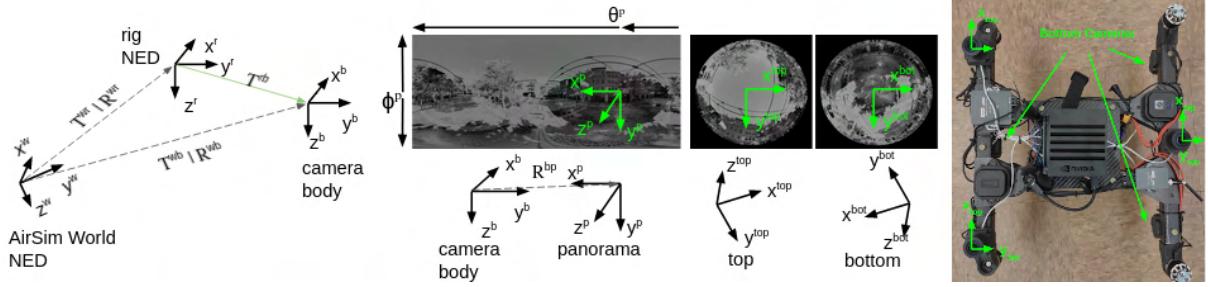


Fig. 3. Coordinate Frame Definitions.

$T_{rb} = R_{wr}^{-1}(T_{wb} - T_{wr})$. Note, this relative translation vector is not constant because Gaussian noise is added to the camera location during data collection.

The fisheye images were created by using the double sphere model to unproject the fisheye coordinates to find the corresponding panorama sampling coordinate [12]. In this work, the fisheye cameras have a FOV of 195° and Kalibr was used to find the corresponding calibration coefficients f_x, f_y, c_x, c_y, ξ , and α [11]. The unprojected camera coordinate vectors are in the panorama coordinate system, as shown in Fig. 3. To be translated into either the bottom camera frame where the z-axis lies along the y-axis of the panorama frame, the unprojected coordinate vectors are rotated along the x-axis by $\frac{\pi}{2}$. If the camera is pointed upwards, a x-axis rotation of $-\frac{\pi}{2}$ is applied before a z-axis rotation of π is applied. These coordinate frame definitions are used to find the corresponding sampling locations from the panorama to the fisheye image. The fisheye images used with Kalibr to calibrate the fisheye cameras were 686-by-686 in resolution, so the resulting sampling locations were scaled to generate a fisheye image of the same resolution to be compliant with the calibration coefficients.

B. Architecture Overview

As shown in Figure 4, the model takes six RGB fisheye camera images and six fisheye-to-panorama sampling grids generated for spherical sweeping as input. The input 686-by-686 RGB images are of shape $[B, 3, H_{in}, W_{in}]$ where B is the batch size and H_{in}, W_{in} are the height and width of the inputs, respectively. The six fisheye-to-panorama sampling grids for spherical sweeping are of shape $[B, N/8, H_{out}, W_{out}, 2]$ where N is the number of depth candidates and H_{out}, W_{out} is the height and width of the output panorama cost volume, respectively. This work uses 192 depth candidates with a minimum distance d_{min} of 0.5 meters and a maximum distance d_{max} of 2^{52} . A shared feature extractor is used by each camera with strided convolution to reduce the feature maps to a final resolution of (W_{in}, H_{in}) . Convolutional layers using residual skip connections are utilized as the basic building block in the feature extractor. A final convolutional layer with bias and without an activation function is used at the end of the feature extraction layers to distinguish between negative features and out-of-FOV regions [3]. This work uses two convolutional layers with a stride of two during

the course of feature extraction, resulting in an output of six feature maps of shape $[B, C, \frac{H_{in}}{4}, \frac{W_{in}}{4}]$ where C is the number of channels outputted by the feature extractor. During spherical sweeping, these feature maps are then warped into the panorama space and swept for all depth candidates. This sweeping process is described more in the Spherical Sweeping section. After the volume of the panoramas for all cameras and for all depth candidates is built, a U-Net is used to compute the final costs. Softargmax is used to find the final depth candidate panorama map.

C. Spherical Sweeping

After the features have been extracted, the feature maps are treated as fisheye images to define a mapping from (H_{in}, W_{in}) to (H_{out}, W_{out}) . In this work, the output shape is chosen as $H_{out} = 176$ and $W_{out} = 320$ to approximately mimic the size of the panorama resulting from the warping of a fisheye image of shape $(\frac{H_{in}}{4}, \frac{W_{in}}{4})$. However, the goal of the network is to output a panorama. To incorporate extrinsic and intrinsic information into the network, the fisheye feature maps are warped into a partial panorama with the double sphere model [12]. This creates six tensors of shape $[B, C, H_{out}, W_{out}]$ for each camera. Instead of regressing the depth directly, which causes blurry depth maps, discretized depth candidates are chosen and a cost volume is built to compute the probability of each depth candidate for the final depth map. The feature maps are warped for all depth candidates, then concatenated together in the channel dimension to create a 4D volume. This 4D volume is to be called the Cost Volume and is of shape $[B, 3C, \frac{N}{S}, H_{out}, W_{out}]$ where N is the total number of depth candidates and S is the step size between spheres during sweeping. This step size represents a memory and computation cost-saving measure whereby only every S depth candidate is used in sweeping in the whole range of depth candidates.

This warping process uses a sampling grid, which is created as input to the network before inference. Using the conventions in Figure 3, an array of spherical rays $R(\theta, \phi, d_i)$ is initialized with eq.1 for each θ, ϕ in the panorama at a specified depth candidate d_i with the NED rig frame as origin.

$$R(\theta, \phi, d_i) = d_i \begin{bmatrix} \sin(\phi) \cos(\theta) \\ -\cos(\phi) \\ -\sin(\phi) \sin(\theta) \end{bmatrix} \quad (1)$$

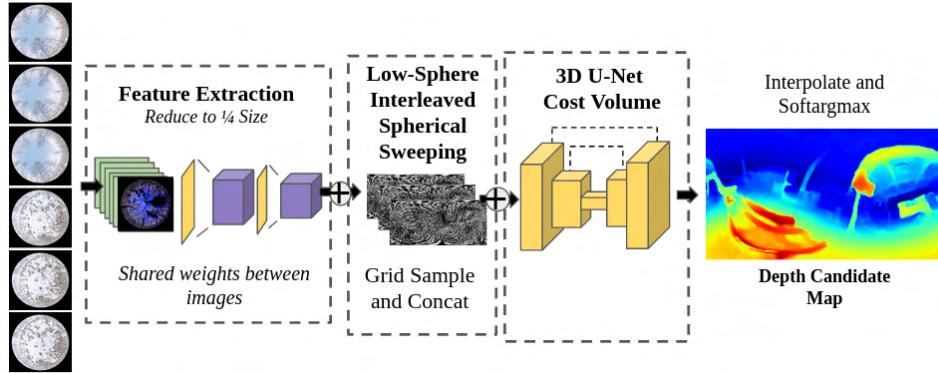


Fig. 4. **Model Architecture.** The six fisheye images use a shared feature extractor where strided convolutions reduces the feature map size to a quarter of the input size. The feature maps warped into the panorama space, then interleaved and concatenated. Finally, the cost volume is computed by a U-Net. Softargmax is applied to the final costs after interpolation to get final depth candidates.

These spherical rays are then transformed and rotated into the top or bottom frame. Finally, the transformed rays are projected into image coordinates (\hat{u}, \hat{v}) that are normalized between $[-1, 1]$ for grid sampling. With this, the sampling locations (\hat{u}, \hat{v}) are found for every (θ, ϕ) in the rig frame for all cameras. This process is repeated for all chosen candidates d_i between (d_{min}, d_{max}) where i is an integer such that $i \in [0, \frac{N}{S}]$.

This work uses two changes to the sweeping process to reduce the memory and computational cost of spherical sweeping. First, the top and bottom partial panoramas are cropped to the shape of $(W_{out}, \frac{H_{out}}{2})$ and interleaved together onto one feature map. This reduces computational and memory costs of the spherical sweeping process by approximately half, yet it disallows any interaction between the top and bottom cameras for depth estimation. Secondly, unlike previous studies, this work uses a larger step size of $S = 8$ to drastically reduce the depth of the cost volume.

D. Cost Volume, Loss Function, and Training

After the fisheye feature maps are spherically swept for all depth candidates, a Cost Volume of shape $[B, 3C, \frac{N}{S}, H_{out}, W_{out}]$ is created. The channel dimension of the input Cost Volume is tripled due to the interleaving of the top and bottom partial panoramas for three pairs of a top and bottom fisheye image. This Cost Volume is passed through a U-Net architecture, treating the depth regression problem as semantic classification. The U-Net outputs a final volume of shape $[B, \frac{N}{S}, H_{out}, W_{out}]$ where each element represents the probability of that pixel being the i th depth candidate where $i \in [0, S, 2S, \dots, N \setminus S]$ where \setminus is the integer division operator such that the remainder is truncated. Softargmax is applied to the final cost volume to find the predicted depth candidate for each pixel.

During training, masked smooth L1 loss is used where the input to the loss function is the predicted and true inverse depth index. The floating point inverse depth index n for a depth candidate d_i is defined by eq.2:

$$n = (N - 1) \frac{d_i^{-1} - d_{max}^{-1}}{d_{min}^{-1} - d_{max}^{-1}} \quad s.t. \quad n \in [0, N - 1] \quad (2)$$

The loss function is masked to exclude losses on pixels where $n > N - 1$, representing a point that is closer to the camera than d_{min} . By masking the output, the network does not learn multiple depths for $n = N - 1$ if the actual depth is closer than the minimum distance. During training, a batch size of seven was used for $S = 8$ while lower batch sizes were used for lower S values due to the large memory cost of sweeping with more spheres. A learning rate of 10^{-3} was assumed with no scheduling.

IV. DATASET

This work also introduces a novel dataset for multiview omnidirectional depth estimation with noisy extrinsics. The data was collected through AirSim, using Unreal Engine (UE) for rendering [13]. The dataset represents a diverse set of real-world conditions. The composition of these conditions is shown in Table I with labels such as Indoors, Outside, Night, Winter, Urban, Industrial, and Nature. The dataset contains 3021 viewpoints from 11 UE environments across 34 different aerial trajectories. Each sample contains a 2048-by-1048 RGB panoramas for each of the 6 camera locations and a 2048-by-1048 ground truth distance panorama rendered at the rig coordinate frame for a total of 7 panoramas per viewpoint. Additionally, the cartesian coordinates and orientation of each camera and the rig frame are annotated. In total, the dataset contains 21,147 panoramas for a storage footprint of 60.8GB.

TABLE I
DATASET CHARACTERISTICS

Category	# of Frames	Category	# of Frames
Urban	1339	Winter	294
Indoors	1092	Industrial	878
Outdoors	1929	Nature	902
Night	531	Total:	3021 Frames

The raw panoramas are saved and then processed into the fisheye images using the given camera intrinsics. This pipeline allows flexibility because the raw panorama contains

TABLE II
INFERENCE TIME COMPARISON

	Input Channels	Runtime (ms)
OmniMVS [3]	8	190
OmniMVS [3]	4	110
Real-Time Sphere Sweeping [10]	-	2.8
(Ours) S=2	8	165.7±27.3
(Ours) S=4	8	67.8 ±6.95
(Ours) S=8	8	41.1±5.76
(Ours) S=8	16	39.3±6.64

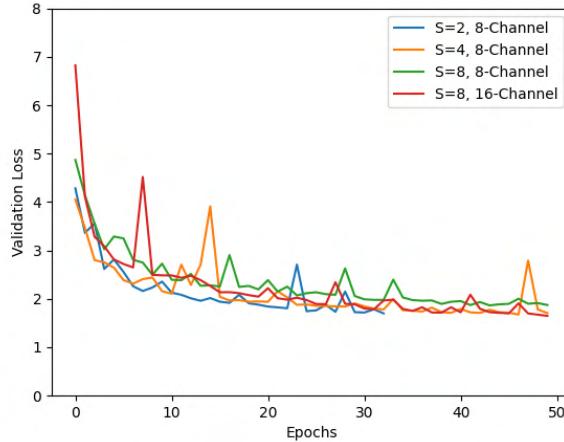


Fig. 5. **Validation Loss History.** The validation losses of our model employing various sphere sweeping steps and output channels over 50 epochs is shown. N -Channel in the legend refers to the number of channels outputted by the feature extractor before sweeping, denoted as C in Sec.3.

all the information needed for produce a fisheye image for an arbitrary orientation and with arbitrary camera intrinsics. While a single panorama at the rig frame could be used to replicate nearby cameras by translating the 3D rays during warping, this does not account for any new occlusions that the cameras would experience as a result of their new location. Lastly, another benefit of collecting raw panoramas is that the source image can be downsampled to replicate fisheye cameras of resolutions lower than or equal to 1024-by-1024 during the warping process, which is a more faithful replication rather than downsampling the fisheye image itself.

V. QUANTITATIVE RESULTS

In this section, the results of training are discussed in a quantitative manner. First, the losses of our model using various spherical sweeping step sizes is presented and discussed. Then, the effect of sweep size on runtime is analyzed and compared with previous work. Lastly, a robustness analysis is shown to validate the durability of the network against changing extrinsics.

As shown in Figure V, the networks all begin to converge after 30 epochs with the last 20 epochs demonstrating an exhaustive training schedule. Note that during a 48 hour training schedule, the $S = 2$ architecture was only able to complete 32 epochs, which was enough to converge. The graph of validation losses shows that while a lower step size helps achieve a lower validation loss, adding more output channels to the feature extractor can reach or outperform models with lower spherical sweeping step sizes. While achieving similar validation losses, the models with higher S values also are much faster as shown in Table II.

By reducing the feature map size by $\frac{1}{4}$ with similar S step sizes, our model had better but not drastically faster inference times compared to previous work. However, as we begin to use higher S values, the inference time drastically improves. With our chosen step size of $S = 8$, we achieve over a four

times faster inference time with comparable validation loss and performance.

In addition to faster runtime, Figure 6 shows that the model is robust against deviations in camera extrinsics. As seen in Figure 6, the best fit line of all samples in the validation set has a slight upward slope of 0.048 loss points per mm of average camera position deviation. This preliminary analysis suggests that training on not only a diverse set of scenes but a diverse set of extrinsics, the resulting model will learn to be robust against measurement errors.

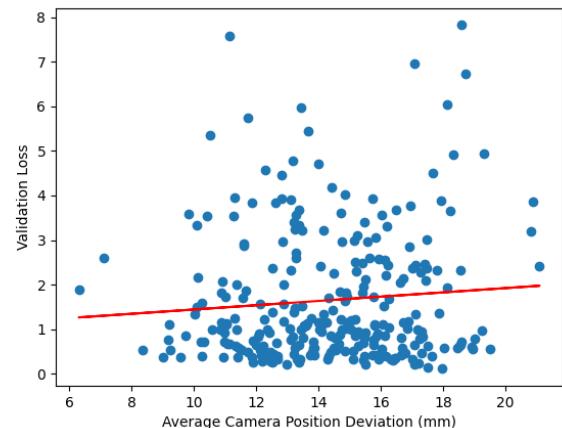


Fig. 6. **Robustness Analysis.** Graph shows a scatter plot of the smooth L1 loss values for each sample in the validation set over the average position deviation of the six cameras relative to the ground truth configuration. The red line is the best fit line of the data.

VI. QUALITATIVE DISCUSSION

In this section, the quality and limitations of the depth panoramas are discussed by showing inferences performed by the model with samples in the validation set. As seen in Figure 7b, the model performs well in a variety of conditions and settings. While edges are generally clear and defined, even more of interest is that individual leaves start to become visible and even the thin lines of the shelving and the lamp pole are defined. However, there are some notable limitations and failures seen in Figure 7. Thin, closely packed lines are often distorted or even non-existent, like those in the fencing or the rails of the panorama that is second from the top. Overall, these preliminary results are impressive and the model can understand a variety of diverse settings. However, more work needs to be done to preserve fine

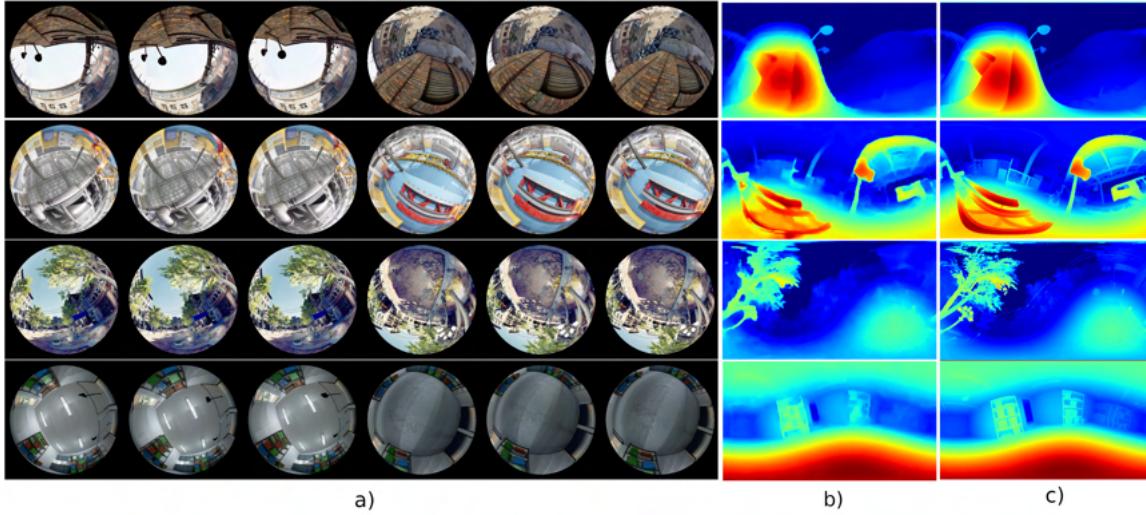


Fig. 7. **Qualitative Results Overview.** Images are predicted from the validation set using the 16-channel model that uses every 8th sphere in spherical sweeping. a) Input Fisheye Images. b) Predicted Depth Panoramas. c) Ground Truth Depth Panorama. Warmer colors indicate depths closer to the rig origin frame.

edges. Additionally, the model needs to be evaluated on environments that are completely unseen to see how the model generalizes.

VII. CONCLUSIONS

In summation, this work presents initial findings for a real-time deep learning model architecture for omnidirectional depth prediction using multiple fisheye images that are facing perpendicular to the scenes' horizon. Additionally, a new dataset of source panoramas is made available that simulates calibration errors by adding noise to the camera positions. More work needs to be done to evaluate the model when trained on a larger dataset. Lastly, data needs to be collected on the comparative performance of SOTA methods when evaluated on samples with known calibration errors.

ACKNOWLEDGMENT

The authors would like to thank the RISS program for providing the opportunity to conduct this research. The authors want to thank Rachel Burcin, Dr. John Dolan, and Jennie Piotrkowski for coordinating RISS. Lastly, the authors want to thank the AirLab for funding the findings published into this working paper, for mentorship throughout the research process, and for providing computation resources.

REFERENCES

- [1] B. F. Spencer, V. Hoskere, and Y. Narazaki, “Advances in computer vision-based civil infrastructure inspection and monitoring,” *Engineering*, vol. 5, no. 2, pp. 199–222, 2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2095809918308130>
- [2] M. Vázquez-Arellano, H. W. Griepentrog, D. Reiser, and D. S. Parafos, “3-d imaging systems for agricultural applications—a review,” *Sensors*, vol. 16, no. 5, 2016. [Online]. Available: <https://www.mdpi.com/1424-8220/16/5/618>
- [3] C. Won, J. Ryu, and J. Lim, “Omnimvs: End-to-end learning for omnidirectional stereo matching,” *CoRR*, vol. abs/1908.06257, 2019. [Online]. Available: <http://arxiv.org/abs/1908.06257>
- [4] R. Komatsu, H. Fujii, Y. Tamura, A. Yamashita, and H. Asama, “360° depth estimation from multiple fisheye images with origami crown representation of icosahedron,” *CoRR*, vol. abs/2007.06891, 2020. [Online]. Available: <https://arxiv.org/abs/2007.06891>
- [5] H. Hirschmuller, “Accurate and efficient stereo processing by semi-global matching and mutual information,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 2, 2005, pp. 807–814 vol. 2.
- [6] J. Chang and Y. Chen, “Pyramid stereo matching network,” *CoRR*, vol. abs/1803.08669, 2018. [Online]. Available: <http://arxiv.org/abs/1803.08669>
- [7] G. Yang, J. Manela, M. Happold, and D. Ramanan, “Hierarchical deep stereo matching on high-resolution images,” *CoRR*, vol. abs/1912.06704, 2019. [Online]. Available: <http://arxiv.org/abs/1912.06704>
- [8] Y. Hu, W. Zhen, and S. A. Scherer, “Deep-learning assisted high-resolution binocular stereo depth reconstruction,” *CoRR*, vol. abs/1912.05012, 2019. [Online]. Available: <http://arxiv.org/abs/1912.05012>
- [9] C. Won, J. Ryu, and J. Lim, “Sweepnet: Wide-baseline omnidirectional depth estimation,” *CoRR*, vol. abs/1902.10904, 2019. [Online]. Available: <http://arxiv.org/abs/1902.10904>
- [10] A. Meuleman, H. Jang, D. S. Jeon, and M. H. Kim, “Real-time sphere sweeping stereo from multiview fisheye images,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.
- [11] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, “Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4304–4311.
- [12] V. Usenko, N. Demmel, and D. Cremers, “The double sphere camera model,” *CoRR*, vol. abs/1807.08957, 2018. [Online]. Available: <http://arxiv.org/abs/1807.08957>
- [13] S. Shah, D. Dey, C. Lovett, and A. Kapoor, “Airsim: High-fidelity visual and physical simulation for autonomous vehicles,” *CoRR*, vol. abs/1705.05065, 2017. [Online]. Available: <http://arxiv.org/abs/1705.05065>
- [14] K. Matzen, M. F. Cohen, B. Evans, J. Kopf, and R. Szeliski, “Low-cost 360 stereo photography and video capture,” *ACM Trans. Graph.*, vol. 36, no. 4, July 2017. [Online]. Available: <https://doi.org/10.1145/3072959.3073645>
- [15] H.-S. Lin, C.-C. Chang, H.-Y. Chang, Y.-Y. Chuang, T.-L. Lin, and M. Ouhyoung, “A low-cost portable polycamera for stereoscopic 360° imaging,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 4, pp. 915–929, 2019.
- [16] M. Menze and A. Geiger, “Object scene flow for autonomous vehicles,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3061–3070.

Detecting and Classifying Waste Bin Garbage Levels Along Transit Bus Routes

Elias Rotondo¹ and Christoph Mertz²

Abstract—Waste bins assist in preventing the spread of trash by serving as central locations where people can discard their garbage. In recent years, researchers have explored using computers to monitor waste bin garbage levels and eliminate the need for human monitoring. Both Internet of Things (IoT) and computer vision technologies have been exploited to accomplish this task. However, IoT approaches require devices to be attached to waste bins which can be costly and time-consuming, and most computer vision methods have not been analyzed in real-world settings. In this paper, we present a waste bin detection and classification system designed for transit buses that utilizes already installed bus cameras to observe the bins. This application is needed because waste bin monitoring systems that rely on humans are ineffective for transportation companies whose bins are spread across large geographical areas. We label bus camera data to create a dataset used to train and evaluate our detection and classification models. Our results show that we can reliably detect waste bins of interest. Moving forward, we plan to complete our system’s pipeline and deploy it on a transit bus to evaluate its live-action performance.

Index Terms—Computer Vision for Transportation, Intelligent Transportation Systems, Object Detection, Segmentation and Categorization

I. INTRODUCTION

It is estimated that the world produces approximately two billion metric tons of trash annually [1]. High levels of municipal solid waste resulting from the consumption of goods, high standards of living, and population growth threaten to pollute the planet if not properly handled [2]. Therefore, it is crucial that communities utilize trash disposal methods, such as using waste bins, to reduce the amount of new garbage introduced to the environment. Today, most waste bins are monitored by humans to determine when they must be emptied. However, this method is inefficient as it becomes increasingly time consuming and tedious for individuals to check on all waste bins when additional ones are deployed. This is especially problematic in cities where waste bins are commonly located on every block. As cities continue to grow and expand, officials will need to develop new methods of monitoring and determining when waste bins located along streets need to be emptied to help reduce the spread of garbage.

We propose accomplishing this job by using the cameras found on public transportation vehicles to monitor street waste bins. In particular, transit buses are well equipped for this task as many have exterior cameras for security

¹Elias Rotondo is with the College of Information and Computer Sciences, University of Massachusetts Amherst, Amherst, MA, USA
erotondo@umass.edu

²Christoph Mertz with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA cmertz@andrew.cmu.edu



Fig. 1. Examples of an empty (left) and a full (right) waste bin found at different bus stops.

purposes and drive along streets picking up and dropping off individuals at bus stops. Additionally, since most bus routes are traversed multiple times throughout the day, waste bins along these routes can be continually monitored. Utilizing buses for this task also benefits the transit companies as they are often responsible for maintaining the waste bins along their routes and therefore an automated monitoring system would reduce their operation costs.

In this paper, we present a waste bin detector that will eventually be deployed onto the *BusEdge* [3] system and used in a garbage level classification pipeline designed to determine when waste bins along bus routes need to be emptied. Using recorded videos captured by a bus’s camera while traveling along its route from Washington, PA to Pittsburgh, PA, we create and label datasets using CVAT [4]. We develop a RetinaNet [5] detector model created using the detectron2 framework [6] and train on our custom dataset. The detector’s success is measured using average precision and recall metrics to determine if the detector can consistently identify waste bins of interest. Based on the collected results, our detector can detect the waste bins we are most interested in with reasonable success. Even though we developed this detector for transit buses, it can easily be deployed on other mobile or fixed cameras that observe waste bins.



Fig. 2. A transit bus similar to the one above was used to collect our training data. Likewise, our finished pipeline will be deployed onto such a bus.

Once completed, a pipeline containing our trained model will be deployed on a transit bus to evaluate its live-action performance. Video frames captured by the bus's camera will be passed to the pipeline using Robot Operating System (ROS) [7]. Waste bin detection using a light-weight model will take place on the bus, and all images with detections will be sent to a cloudlet server to be processed once more by an increased performance detector. Once the system is confident that a waste bin has been detected, the classifier will determine if the bin must be emptied or needs to be attended to. Whenever a waste bin is identified as needing to be emptied, a notification will be generated and sent out to the appropriate parties.

The structure of this paper is as follows: In Section II, a literature review is given on previous works surrounding detecting waste bins and monitoring their garbage levels. Section III provides an overview of the proposed system. The conducted experiments are described in Section IV, and their results presented in Section V. Lastly, a conclusion is provided in Section VI and future works are discussed.

II. RELATED WORK

A. Waste Bin Detection and Garbage Level Classification

Over the years, researchers have proposed using different techniques to create autonomous waste bin garbage level monitoring systems [8]–[11]. Existing works can be divided into two groups based on the underlying technologies: Internet of Things (IoT) and computer vision.

IoT methods address the waste level monitoring problem by attaching IoT sensors to the waste bins [8], [9]. These sensors monitor the fullness of waste bins and report back to central monitoring systems. Different types of sensors can be utilized to determine how full the bins are. For example, Y. Zhao et al. [8] developed sensors that monitor how full waste bins are by sensing changes in the bins' motor-induced vibrations. Alternatively, the sensors presented by S. J. Ramson et al. [9] rely on ultrasonic waves to detect what garbage level has been reached within the bins.

While IoT sensors have proven to be effective at monitoring waste levels and are commercially available, they are not

without their faults. Each waste bin requires its own sensor, making such an approach expensive when install these IoT devices to a large number of bins. Additionally, as noted by Y. Zhao et al. [8], sensors can become damaged or knocked off after deployment. This greatly increases the difficulty of maintaining such a system as someone must be prepared to replace the sensors when necessary.

Computer vision algorithms have also been explored to monitor the garbage level in waste bins [10], [11]. Such algorithms are developed by extracting a set of key features from images in a training dataset that are then used to train a classifier. In [10], M. A. Hannan et al. propose two waste level classifiers that extract the gray level aura matrix of an images before using a multilayer perceptron or K-nearest neighbor classifier to determine how full the waste bin is. F. Aziz et al. [11] take a different approach. First, the waste bin opening is located using the Canny Edge Operator followed by applying the Hough transform. A Gabor filter is then applied to the image to extract additional properties. Using the collected features, F. Aziz et al. then pass them to a support vector machine or multilayer perceptron classifier that will place the image into one of three classes: empty, partially full, or full [11].

As was the case with the IoT approaches, computer vision algorithms still need to be improved and further explored before they can be fully exploited for garbage level monitoring. Such algorithms require large amounts of training data to properly and reliably function as intended. Creating appropriate datasets using real-world data is a time consuming and often costly endeavor. To our knowledge, no such datasets are publicly available for detecting waste bins and classifying them based on their garbage levels. While both [10] and [11] show promising results, these algorithms were trained and evaluated using artificial datasets. As a result, there is no guarantee that these approaches will work in real-world environments, and so their performance is still largely unknown from a practical application standpoint.

B. Deep Learning

In more recent years, computer vision research has heavily shifted to using deep learning methods. Within the area of waste management, many researchers have successfully utilized deep learning algorithms to detect, segment, or classify garbage and garbage-related objects in a wide arrangement of environments [12]–[15]. Deep learning techniques are also readily available to researchers through the use of frameworks such as detectron2 [6]. One advantage of using these frameworks is that they often provide common models that are already pre-trained using popular datasets such as ImageNet [16] and COCO [17] and therefore only need to be fine-tuned.

Today, there exists many different object detection models including RetinaNet [5], Faster R-CNN [18], R-FCN [19], SSD [20], and YOLO [21]. Convolutional neural networks (CNNs) approaches such as these have been shown to produce high performing models for various object detection tasks [22]. When selecting which architecture to use,

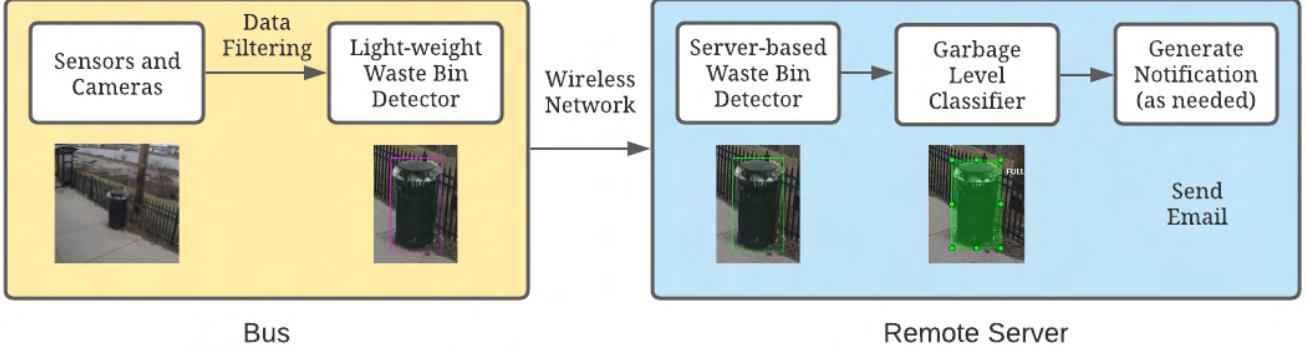


Fig. 3. Waste bin detection and classification pipeline.

researchers need to take into account model accuracy and available resources as different approaches balance these requirements differently and often put a heavier emphasis on one over the other. Additionally, feature extractor methods such as ResNet50 [23] and MobileNet_V1 [24] can be used to further improve the performance of object detection models.

C. Vehicle Edge Computing

As the automotive industry works to develop connected vehicle systems, edge computing will play a crucial role in balancing the need for real-time computing within vehicles while allowing for some tasks to be offloaded to remote servers [25]. While vehicle edge computer poses many challenges, researchers such as X. Xu et al. [26] are working to ensure that edge computing transmitting tasks can take place without compromising the safety or functionality of vehicles. One example of a system that utilizes vehicle edge computing is proposed by C. Ye [3]. The system, referred to as *BusEdge*, runs on public transit buses and provides a platform for applications to access bus data in real-time such as camera video streams and the bus's latitude and longitude coordinates.

III. SYSTEM OVERVIEW

In this section, we discuss the components that make up our proposed system which was inspired by the pipeline proposed in [3]. Fig. 3 visualizes how the individual pieces are connected and provides an example of the output or triggered response after each segment is finished running.



Fig. 4. Hardware components located on the bus (from left to right: onboard computer, exterior camera, GPS and network antenna).

A. Bus Sensors and Data Pre-Processing

The transit bus is equipped with a series of sensors including multiple exterior cameras and a global positioning system (GPS). The sensors communicate with the bus's onboard computer using ROS [7] through individual topic publish-subscription channels between each sensor and the computer. For our application, we utilize the exterior camera on the right side of the bus angled towards the front of the bus and the GPS. Latitude and longitude coordinates are encoded into the individual frames from the camera's video stream. These images are then continually processed through a filter before being sent to the onboard waste bin detector. This filter removes frames so that the overall frames-per-second rate is lowered to ensure that the light-weight detector can process new frames in real-time without creating a bottleneck. Several components belonging to the bus's hardware system can be seen in Fig. 4.

B. Waste Bin Detection

Within the pipeline, there are two main constraints we must address relating to the transit bus. The first is that the bus is only equipped with a basic computer system that has limited memory and no graphics processing unit. As a result, the applications running on the bus must require only minimal resources. The second limitation with using the bus is that there only exists a finite amount of bandwidth to be used when communicating between the bus's onboard system and the cloudlet server. Even after undergoing pre-processing, it is infeasible to transfer all the data collected by the bus's sensors to the cloudlet. Therefore, we need to further reduce the number of video frames by using a model to identify the frames of interest within the waste bin detection task.

To resolve these constraints, we propose the use of two waste bin detection models within our pipeline. The transit bus's onboard computer will run a lightweight model, such as an SSD [20], to detect waste bins as the bus drives past them. While this simpler detector will have a lower performance compared to models that consist of more complex neural networks, it will significantly reduce the overhead on the bus's internal system. Additionally, this model can be tuned

to be more lenient in identifying waste bins even if it results in more false positives as these instances will be eliminated later on the servers where more computational power is available. Since most waste bins of interest appear in multiple frames, this detector will also have multiple chances to identify instances as the bus drives past them. Once this model detects a waste bin instance in the current frame, the system will send it to the cloudlet through the edge network.

Once the cloudlet server receives a frame from the bus, it will pass the image to another detection model which has more computing resources at its disposal. This detector will use a more complex architecture, such as RetinaNet [5], therefore enabling it to better recognize waste bins when compared against the bus's lightweight model. Additionally, once a waste bin is detected by the cloudlet model, it will crop the region containing the bin from the original frame before passing it to the garbage level classifier. This functionality eliminates the need for the classifier to locate the waste bin of interest before assigning it to a class.

C. Garbage Level Classification

The final model within the pipeline will be the classifier used to determine if a detected waste bin needs to be emptied. To reduce the number of classes within the model, all waste bins with no visible trash will be considered not full and any bin with visible trash will be classified as full regardless of how much trash is present. Additionally, the classifier will also determine if there are any garbage bags surrounding the waste bins as this signals that someone still needs to attend these bins. Similar to the second detector, the garbage level classifier will use a model that focuses on performance such as a RetinaNet [5] or Faster R-CNN [18]. When a full waste bin is detected, the information will be passed to the last component within the pipeline responsible for notifying the appropriate parties.

D. Full Waste Bin Response

The last piece of our proposed pipeline is responsible for informing the transit bus company or associated party that there is a waste bin that needs to be attended to. After receiving a full classification instance, this system component will decode the GPS location associated with the instance to determine if the detected and classified waste bin is within the jurisdiction of the overseeing parties. Once this information is confirmed, an automatic email containing the image associated with the full instance and its GPS location will be sent to the necessary individuals. Additionally, this information will be used to update an interactive webpage that maps out the bus route so that the locations of all full waste bins can be found in one centralized location. The webpage will also contain the images of the full instances so that the involved party members can confirm that the bins are truly full before sending someone to collect the garbage in the event that a false positive occurred.

IV. EXPERIMENTATION

A. Dataset

To evaluate the performance of our models, we created custom datasets using recorded video streams obtained during normal operations. Annotations were manually added to the datasets through the use of CVAT [4]. Bounding boxes were placed over instances of waste bins in the detection dataset, and three classes were identified for the classification dataset: full, not full, and garbage bag besides a waste bin. For training purposes, both datasets were split by a 60-20-20 ratio for the training, validation, and test sets, respectively. All images used in the training set were taken from the same date. The images used in the validation and test sets come from the same run taken at a different date than the training set, although images are only used in one set and not the other. Tables I and II provide the distribution of samples in each subset. For grouping purposes, small refers to a bounding box area less than 1,024 pixels, medium refers to a bounding box area between 1,024 and 9,216 pixels, and large refers to a bounding box area greater than 9,216 pixels. Our datasets are publicly available and can be accessed through the kaggle website [27], [28].

TABLE I
DETECTION DATASET - WASTE BIN DISTRIBUTION AMONGST THE TRAINING, VALIDATION, AND TEST SETS

Bounding Box Size	Training	Validation	Test
Small	818	268	268
Medium	604	247	247
Large	141	72	71
All Sizes	1563	587	586

TABLE II
CLASSIFICATION DATASET - CLASS DISTRIBUTIONS AMONGST THE TRAINING, VALIDATION, AND TEST SETS

Class	Bounding Box Size	Training	Validation	Test
Not Full Waste Bins	Small	767	265	268
	Medium	518	241	239
	Large	116	65	66
	All Sizes	1401	571	573
Full Waste Bins	Small	53	2	1
	Medium	87	7	6
	Large	26	7	5
	All Sizes	166	16	12
Garbage Bags	Small	54	31	31
	Medium	61	42	47
	Large	19	6	4
	All Sizes	134	79	82

B. Model Training and Evaluation

Within the scope of this paper, two experiments will be analyzed. The first evaluates the performance of a RetinaNet [5] waste bin detector similar to the final detector model that will run on the cloudlet server. The second experiment provides a proof-of-concept test to determine if a single model can be used to both detect and classify full waste bins. Our selected model for this proof-of-concept experiment is a RetinaNet [5] multi-class detector. Both models are created using the detectron2 framework and start with pre-trained RetinaNets [5] that are then trained on their respective datasets as previously described. Each model is trained for 20,000 iterations on the training dataset with periodic evaluations on the validation set every 1,000 iterations. After the training procedure is completed, both models are evaluated on their respective test sets using the standard detectron2 COCO evaluator and their metric scores are recorded. These results are presented in the following section.

V. RESULTS

A. Server-Based Waste Bin Detector

After training our waste bin detector as defined in the previous section, our model achieves an overall average precision of 44.2% as computed by the detectron2 COCO evaluator when assessed on the test set. This score increases to 67.4% when the Intersection over Union (IoU) metric is restricted to 0.5. Additionally, the model yields an average recall of 56.9%. However, we see that these scores increase when we restrict the evaluation to take into account different bounding box area sizes of detected waste bins. For example, our model reaches an average precision of 57.3% and an average recall of 67.1% when detecting medium-sized waste bins. These scores further increase to AP = 74.8% and AR = 79.2% when only considering the large-sized waste bin detections. Figs. 5 and 6 show the precision-recall curves for the medium and large-sized waste bins. Based on the results, there is still room for improvement in training the server-based waste bin detector. However, the average precision and recall metrics for large-sized waste bins confirm that our current detector can reasonably identify bins of interest since most fall under the large-sized category in at least one video frame. This is further supported by the precision-recall curve for large-sized waste bins.

TABLE III
SERVER-BASED WASTE BIN DETECTOR METRICS

Bounding Box Size	Average Precision	Average Recall
All Sizes	44.2%	56.9%
Medium	57.3%	67.1%
Large	74.8%	79.2%

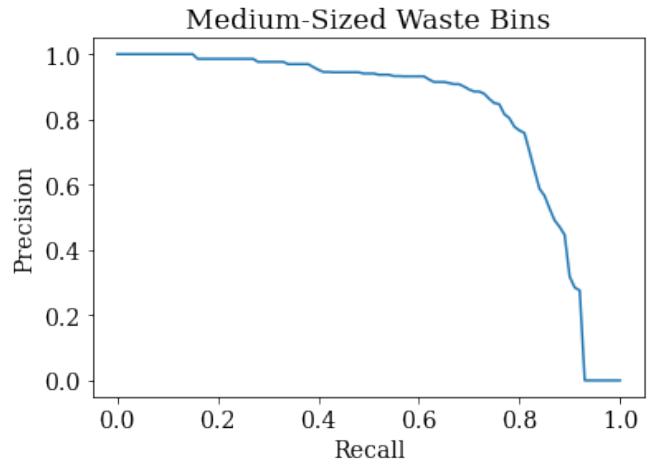


Fig. 5. Precision-recall curve for medium-sized waste bins (bounding box area is between 1,024 and 9,216 pixels).

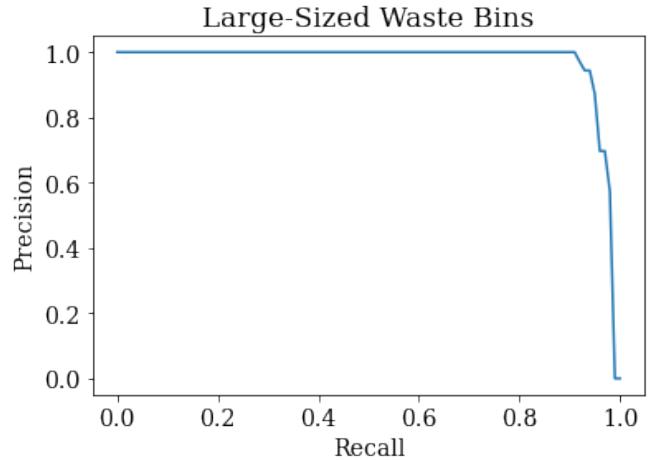


Fig. 6. Precision-recall curve for large-sized waste bins (bounding box area is greater than 9,216 pixels).

B. Proof-of-Concept Waste Bin Multi-Class Detector

After following the training procedure detailed in Section IV, our waste bin multi-class detector was evaluated using the classification test set. For each class, the average precision score was recorded. The model achieves an average precision of 45.3% for not-full waste bins, 7.0% for full waste bins, and 54.0% for garbage bags. These metrics signal that the multi-class detection approach is insufficient for completing our desired task. Moving forward, we will separate the detection and classification tasks from one another to determine if using separate but connected models improves the final classification performance.

VI. CONCLUSIONS

In this paper, we present a waste bin detection and garbage level classification system that can be installed onto public transit buses to monitor waste bins along bus routes. Our results support that we can train detection models to identify waste bins as the bus drives past them. In particular, our

evaluated model frequently detects waste bins of interest since these instances are always medium or large-sized within the captured images. Additionally, important waste bin instances are usually captured in several frames, further increasing our system's capabilities of detecting all bins of interest. This paper also explores the possibility of using a single model to both detect and classify full waste bins. However, the results do not support that such a model can successfully accomplish this task.

A. Future Works

Before advancing with the development of our proposed pipeline, we will gather and label more bus data. While all parts of the system will benefit from having access to more training data, the classification of full waste bins will improve the most as our dataset currently has very few instances belonging to this class. The data will be collected along the same bus route using the same transit bus company to keep data collection consistent.

Upon completion of our pipeline, we will deploy the respective models onto the transit bus and cloudlet server to evaluate their performance in real-time. Once we have collected enough results after deployment, we will re-analyze our pipeline and make appropriate changes until it can detect and classify full waste bins with high consistency. We will then deploy the system onto other buses traveling along different routes within the same area to evaluate how the system performs in environments independent of the training data. We hope our work will inspire other researchers to develop similar useful systems that utilize public transportation to accomplish their tasks.

ACKNOWLEDGMENTS

This research has been supported by the National Science Foundation under Grant No. 1659774. We thank the Robotics Institute Summer Scholars program for making this experience possible and for providing feedback on this paper. Additionally, we would like to thank Canbo Ye and Tom Bu for the guidance they provided.

REFERENCES

- [1] S. Kaza, L. Yao, P. Bhada-Tata, and F. Van Woerden, "What a waste 2.0 : A global snapshot of solid waste management to 2050," *World Bank Publications*, 2018.
- [2] S. Nanda and F. Berruti, "Municipal solid waste management and landfilling technologies: a review," *Environmental Chemistry Letters*, vol. 19, no. 2, pp. 1433–1456, 2021.
- [3] C. Ye, "Busedge: Efficient live video analytics for transit buses via edge computing," Carnegie Mellon University Robotics Institute Technical Report 21-46, 2021.
- [4] Intel, "Computer vision annotation tool," 2018. [Online]. Available: <https://github.com/openvinotoolkit/cvat>
- [5] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.
- [6] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," 2019. [Online]. Available: <https://github.com/facebookresearch/detectron2>
- [7] Stanford Artificial Intelligence Laboratory et al., "Robotic operating system." [Online]. Available: <https://www.ros.org>
- [8] Y. Zhao, S. Yao, S. Li, S. Hu, H. Shao, and T. F. Abdelzaher, "Vibebin: A vibration-based waste bin level detection system," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, vol. 1, no. 3, Sept. 2017. [Online]. Available: <https://doi.org/10.1145/3132027>
- [9] S. J. Ramson, D. J. Moni, S. Vishnu, T. Anagnostopoulos, A. A. Kirubaraj, and X. Fan, "An iot-based bin level monitoring system for solid waste management," *Journal of Material Cycles and Waste Management*, vol. 23, no. 2, pp. 516–525, 2021.
- [10] M. A. Hannan, M. Arebey, R. A. Begum, and H. Basri, "An automated solid waste bin level detection system using a gray level aura matrix," *Waste management*, vol. 32, no. 12, pp. 2229–2238, 2012.
- [11] F. Aziz, H. Arof, N. Mokhtar, M. Mubin, and M. S. A. Talip, "Rotation invariant bin detection and solid waste level classification," *Measurement*, vol. 65, pp. 19–28, 2015.
- [12] M. Yang and G. Thung, "Classification of trash for recyclability status," *CS229 Project Report*, vol. 2016, 2016.
- [13] Y. Wang and X. Zhang, "Autonomous garbage detection for intelligent urban management," in *MATEC Web of Conferences*, vol. 232. EDP Sciences, 2018, p. 01056.
- [14] B. De Carolis, F. Ladogana, and N. Macchiarulo, "Yolo trashnet: Garbage detection in video streams," in *2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*. IEEE, 2020, pp. 1–7.
- [15] J. Hong, M. Fulton, and J. Sattar, "Trashcan: A semantically-segmented dataset towards visual detection of marine debris," *arXiv preprint arXiv:2007.08097*, 2020.
- [16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.
- [17] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: <http://arxiv.org/abs/1405.0312>
- [18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.
- [19] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in neural information processing systems*, 2016, pp. 379–387.
- [20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.
- [21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
- [22] L. Chen, S. Lin, X. Lu, D. Cao, H. Wu, C. Guo, C. Liu, and F.-Y. Wang, "Deep neural network based vehicle and pedestrian detection for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.
- [23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [24] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.
- [25] D. Grewe, M. Wagner, M. Arumaithurai, I. Psaras, and D. Kutscher, "Information-centric mobile edge computing for connected vehicle environments: Challenges and research directions," in *Proceedings of the Workshop on Mobile Edge Communications*, ser. MECOMM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 7–12. [Online]. Available: <https://doi.org/10.1145/3098208.3098210>
- [26] X. Xu, Y. Xue, X. Li, L. Qi, and S. Wan, "A computation offloading method for edge computing with vehicle-to-everything," *IEEE Access*, vol. 7, pp. 131 068–131 077, 2019.
- [27] E. Rotondo, C. Mertz, and C. Ye, "Waste bin detection dataset (riss2021)," <https://www.kaggle.com/elirotondo/waste-bin-detection-dataset-riss-2021>, 2021.
- [28] ———, "Waste bin multi-class detection dataset (riss2021)," <https://www.kaggle.com/elirotondo/waste-bin-multiclass-detection-dataset-riss2021>, 2021.

Generalization in Reinforcement Learning through Representation

Rutav Shah¹ and Vikash Kumar² and Yunzhu Li⁴ and Abhinav Gupta^{2 3}

Abstract—Developing agents that are capable of acquiring complex behaviors which remain effective even in an uninstrumented environment (like home, hospital etc) can lead to generalist robots. Such agents can be used outside the laboratory conditions in providing care and enhancing productivity. Such uninstrumented settings demands learning directly from robot's onboard sensors like camera, joint encoders etc which poses the high dimensionality issue. Most of the current approaches get away by learning narrow, task specific, low-dimensional representations. These narrow representations lack semantic understanding which inhibits its generalization capabilities. To this end, we propose DDRL which learns policy that is robust to large intra-category variations like semantically similar tools (e.g spatula and spoon). In contrast to previous approaches, the learned representations ignores task irrelevant details while remaining effective to such intra-category variations. DDRL trained on a set of tools (knife, spatula, ratchet, screw driver, turned) and generalizes to new unseen objects e.g hammer without any additional efforts.

Index Terms—Robotics, Reinforcement Learning, Representation Learning

I. INTRODUCTION

Reinforcement Learning has seen tremendous progress over a wide range of applications like games [1]–[4], finance [5] [6], healthcare [7] [8], robotics [9] [10] and many more. Recent success stories in robotics are not only limited to simulation [11]–[13] but also real-world successes [14]–[16] have a fair share. However, most real-world stories that acquire complex manipulation skills are still limited to a laboratory or in a highly instrumented setting mainly due to (a) high sample complexity which makes it infeasible to learn in the real world. (b) lack of direct access to the underlying state information which hence must be extracted from on board sensors. (c) high variability in real-world, for e.g change in lightning conditions, background etc. These limitations restrains the applicability of reinforcement learning in robotics in a more unstructured, real-world setting. This work takes a small stride in this direction.

Learning without environment instrumentation in a more realistic setting can be quite challenging for robots [17] , however, animals especially humans learn behaviors quite efficiently and adapt very easily to the changes in the surrounding. Two important factors that contributes to the

success (a) Efficiency - Humans acquire behaviors efficiently while learning directly from the sensory information like vision, touch etc. Such efficient learning can be attributed to the representation of the information learned by humans [18]. Most of the recent works [19] [20] focus on this aspect of learning representation that efficiently solves the task in hand. (b) Generalization capability - Unlike machines, humans generalize to unseen objects/scenes very easily. We can recognize an object even when seen under strikingly different conditions [21]. This is due to the fact that we have a very good semantic understanding of objects for e.g while picking up a mug we inherently segment out the handle from rest of the body irrespective of the shape/size of the handle. However, very few works have focused on learning semantically meaningful representations [22] in order to bring out the overall generalization capability. Learning narrow, task-specific representation leads to sample efficient solution but are quite brittle and break easily when subjected to small variations in surroundings [23] [24] which hinders the generalization performance.

The desiderata of a good representation in reinforcement learning [25] can be summarized as follows (a) should be low dimensional for sample efficient learning of policy. (b) pays attention to the relevant part of the tasks and captures all the necessary information (c) ignores the irrelevant information like background, clutter etc (d) robust to the variations in the environment which enhances the generalization performance. (e) semantically meaningful representation that remains consistent across intra-class variations. Towards this goal, we propose DDRL that learns semantically consistent representation across different instances of objects belonging to the same class while being sample efficient.

Our contributions : We list our major contributions :

- We present a simple framework that combines the progress of computer vision and learning representation in reinforcement learning to provide a robust, sample efficient solution with better generalization performance than prior methods.
- Benchmark the sample efficiency of the policy learned on a complex, dexterous manipulation ADROIT benchmark [13] and show competitive performance to state of the art approaches.
- Investigate the performance of the representation learned when subjected to various variations like change in camera position, background, introduction of a new object into the scene.

¹Rutav Shah is with the Computer Science and Engineering Department, Indian Institute of Technology Kharagpur rutavms@gmail.com

²Vikash Kumar and Abhinav Gupta are with the Facebook AI Research, USA vikashplus@gmail.com

³Abhinav Gupta is with the Robotics Institute, Carnegie Mellon University, USA gabhinav@andrew.cmu.edu

⁴Yunzhu Li is with the Computer Science & Artificial Intelligence Lab, Massachusetts Institute of Technology, USA liyunzhu@mit.edu

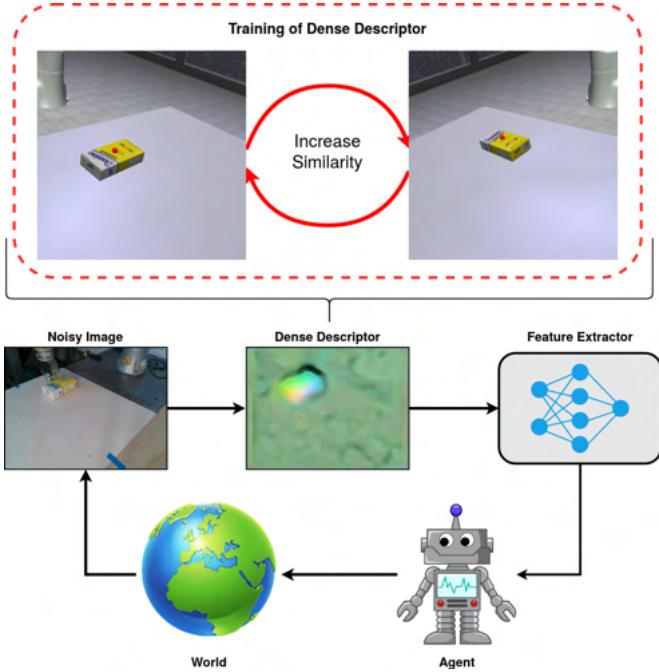


Fig. 1: DDRL combines the progress from Representation learning in Reinforcement learning and Computer Vision; provides a framework that learns representation robust to various distractions and generalizes across new unseen objects in a zero shot fashion. It learns a dense descriptor by bringing the descriptor vectors corresponding to the same position viewed from different camera angles closer, highlighted in RED (Refer to Eq. 8). During the policy learning phase, the dense descriptor feature map is passed through a feature extractor to get a low-dimensional, compact representation which in turn is used as a representative of the environment state.

II. RELATED WORKS

A naive approach to learning from visual observations is to learn directly in the pixel space without explicit distinction between the feature encoder and policy. The seminal work from [2] masters Atari games uses only raw image pixels. Other works that successfully learns policies in real-world directly from images include [26] [27]. Another recent work that has shown success in real world [16] which uses the sample-efficient off policy algorithm SAC [28] directly in the pixel space. These approaches, however, require large number of samples to extract the useful information from images which makes it infeasible to learn outside the laboratory conditions. Our work learns an explicit feature representation and then learns the policy in the feature space, this reduces the sample complexity drastically.

Another technique is learn explicitly learn key-points from images that can be used as a representative of the underlying state information to solve the task [22] [29] [30] and more recently [31]. [32] [33] learn a dense descriptor of the input image and extract keypoints from the descriptor space. Although these methods successfully learn low-dimensional, compact representation of the images, learning task specific keypoints makes it less robust to variations in the environ-

ment and difficult to scale with increasing complexity in the task.

A different line of work is to learn feature representations from the pixel space and use the extracted features as a substitute of exact state of the environment. Works like [34] [35] [17] [36] learn the representations using variational inference techniques, while some [37] [19] use a contrastive objective along with the RL objective to learn useful features. [20] [38] make use of image augmentations and demonstrate state comparable results on Deep Mind Control suite [39], Atari [2]. Learning task specific representation in tandem with the RL objective leads to the issue of non-stationarity since the policy is dependent on the representation while the representation depends on the distribution induced by the policy. In order to avoid this issue, we learn the representation beforehand and freeze the parameters throughout the training of agent.

III. METHODOLOGY

DDRL solves the manipulation tasks by combining three major components : (a) Learning a robust dense descriptor which ensures semantic consistency across the different variations introduced in the scene. (Section III-A) (b) Learning a compact low dimensional representation of the descriptor space eliminating the *curse of dimensionality* for the policy learning. (Section III-B) (c) Finally, a sample efficient policy that solves the task using the obtained representations. (Section III-C)

A. Learning a Dense Descriptor

Goal is to learn a visual representation of the input image that i) ignores the task irrelevant details ii) robust to environmental variations like lightning conditions iii) remains semantically consistent when subjected to intra class variations. Following the success of [32], [33], [40], [41], we learn a dense descriptor using the learning strategy proposed in [33]. The objective of the dense descriptor model ($f_{\theta_{dd}}(\cdot)$) is to map a full resolution RGB image ($\mathcal{R}^{W \times H \times 3}$) to a dense descriptor space ($\mathcal{R}^{W \times H \times D}$), i.e, $f_{\theta_{dd}}(I) : \mathcal{R}^{W \times H \times 3} \rightarrow \mathcal{R}^{W \times H \times D}$ where I represents the input image and each pixel in the input image is mapped to a dense descriptor vector of dimension D . The learning objective is to bring the descriptor vectors corresponding to the same position (in the actual world) viewed from different camera angles closer for example in Figure 1 highlighted in RED. Let I_a, I_b represent images of the same scene captured from two different camera angles, $f_{\theta_{dd}}(I_a)(u_a)$ represents the descriptor vector of the image I_a corresponding to the pixel location u_a . We desire to bring the descriptor vector - $f_{\theta_{dd}}(I_a)(u_a)$ and $f_{\theta_{dd}}(I_b)(u_b)$ where u_a, u_b represent the pixel co-ordinates in images I_a, I_b of the same point in the world frame. In order to achieve this, two loss components are used [32], [33],

(1) **Heatmap Loss** : Let p be any pixel space location in the image I_b , The ground truth heatmap with respect to the pixel location u_b can be defined as

$$H_{u_b}^*(p) = \exp\left(-\frac{\|p - u_b\|_2^2}{\sigma^2}\right) \quad (1)$$

A predicted heatmap is obtained using the descriptor vector $d^* = f_{\theta_{dd}}(I_a)(u_a)$ as the reference descriptor as follows :

$$\hat{H}_{u_b}(p) = \exp\left(-\frac{\|f_{\theta_{dd}}(I_b)(p) - d^*\|_2^2}{\eta^2}\right) \quad (2)$$

Now, a simple Mean Squared Error loss (MSE) is applied between the ground truth (H^*) and predicted (\hat{H}) heatmap.

$$L_{heatmap} = \frac{1}{|\Omega|} \sum_{p \in \Omega} \left\| \hat{H}(p) - H^*(p) \right\|_2^2 \quad (3)$$

(2) **Spatial Expectation Loss** : We have the predicted heatmap from Equation 2. This can be normalized to obtain the probability distribution over the image I_b as

$$\tilde{H}_{u_b}(p) = \frac{\hat{H}_{u_b}(p)}{\sum_{p' \in \Omega} \hat{H}_{u_b}(p')} \quad (4)$$

So the predicted expected pixel location can be calculated using the probability distribution :

$$J_{pixel}(u_b) = \sum_{p \in \Omega} p * \tilde{H}_{u_b}(p) \quad (5)$$

Given the depth map Z , the predicted expected depth can be calculated

$$J_{depth}(u_b) = \sum_{p \in \Omega} Z(p) * \tilde{H}_{u_b}(p) \quad (6)$$

The spatial expectation is calculated using the ground pixel location u_b and depth value $Z(u_b)$

$$L_{spatial} = \|u_b - J_{pixel}(u_b)\| + \|Z(u_b) - J_{depth}(u_b)\| \quad (7)$$

Total Loss : The total loss is a weighted combinations of both the terms

$$L = w_{heatmap} * L_{heatmap} + w_{spatial} * L_{spatial} \quad (8)$$

As observed in [40], the matches (u_a, u_b) are sampled only from the object which can be done using object masking [42]¹. Once the dense descriptor is trained using the loss 8, the weights are frozen throughout the training of the RL agent.

B. Low-dimensional representation

The dense descriptor obtained using $f_{\theta_{dd}}(\cdot)$ is quite high dimensional $W \times H \times D$, where in our setting $H = 480$, $W = 640$ and $D = 3$. Such high dimensional input leads to sample inefficiency during policy learning [19], [20], [38], [43], which is undesirable. Thus, following [43], we use a model pre-trained on a large class of real world samples as a fixed feature extractor. Formally, let $h(\cdot)$ be the fixed feature extractor, $h : \mathcal{R}^{W \times H \times D} \rightarrow \mathcal{R}^d$ where $W \times H \times D \gg d$. We use the standard Resnet model [44] pre-trained on ImageNet dataset [45] to extract the compact, low-dimensional representation from the descriptor space. Although, our method is agnostic to the policy learning method used to acquire the policy, we describe in brief the base Reinforcement learning algorithm used in the next section.

¹since we use a simulator to benchmark the performance object masking can be done easily.

C. Policy Learning

The encoded representations obtained from the input image using the dense descriptor and feature extractor as discussed in Section III-A and III-B are used as a representative of the state information which is not readily available in real-world. DDRL is built upon an on-policy algorithm NPG [46] owing to its stability and effectiveness in solving complex problems. Additionally, in complex environments we augment the data with few human demonstrations as in [13] that helps in (i) relaxing the domain expertise required in reward shaping (ii) solving the early exploration issue (iii) leading to more sample efficient solution. It is important to note that our approach is agnostic to the policy learning algorithm and in principle any base Reinforcement learning method can be used. The encoded representations obtained from the input image using the dense descriptor and feature extractor as discussed in Section III-A and III-B are used as a representative of the state information which is not readily available in real-world. DDRL is built upon a on-policy algorithm NPG [46] owing to its stability and effectiveness in solving the tasks.

IV. EXPERIMENTAL EVALUATIONS

Our experimental evaluations aims to address the following points, (a) Compare the sample efficiency of our approach with state of the art methods on a complex dexterous manipulation benchmark, (b) How robust is the learned descriptor when exposed to novel objects (c) show some elementary results on the generalization performance of the policy using our approach when exposed to unseen objects.

A. Benchmarking

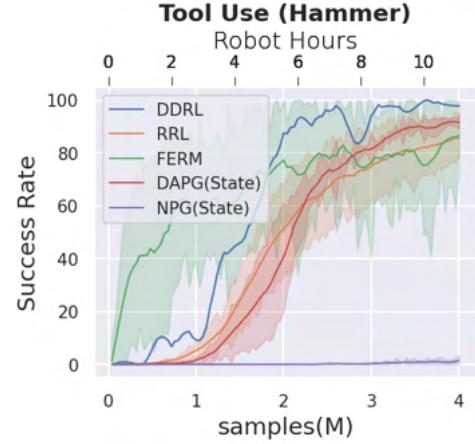


Fig. 2: Performance on ADROIT manipulation benchmark [13] : Natural Policy Gradient [46] fails to make any progress in the task which suggests the difficulty of the task suite. DAPG(State) [13] learns to solve the task with the help of few demonstrations, however, it assumes access to the underlying state information. DDRL quickly learns to solve the task outperforming the SOTA approaches FERM [47], RRL [43], which helps us evaluate the sample efficiency of our method.

In Figure 2, we benchmark the performance of DDRL on the high dimensional dexterous manipulation benchmark [13] in comparison to various state of the art methods. The Natural Policy Gradient (NPG(State)) [46] [in Purple] even with the privileged state information fails to make progress in the task. This establishes the difficulty of our task suite. Furthermore, DAPG(State) [13] [in RED] leverages a few demonstrations to bootstrap the learning process and achieves success using the privileged state information. However, such information is not readily available when learning in real-world.

DDRL [in BLUE] demonstrates better performance even when compared to DAPG(State) [in RED] which can be considered as an oracle of our method since it has access to the underlying state information. Additionally, it also outperforms RRL [43] [in ORANGE] that directly uses Resnet-34 features and uses similar reinforcement learning framework. For fair comparison, we keep the hyper-parameters same as mentioned in RRL. FERM [47] is another baseline that uses off-policy algorithm SAC [28] which shows competitive performance, however, is quite unstable and saturates early.

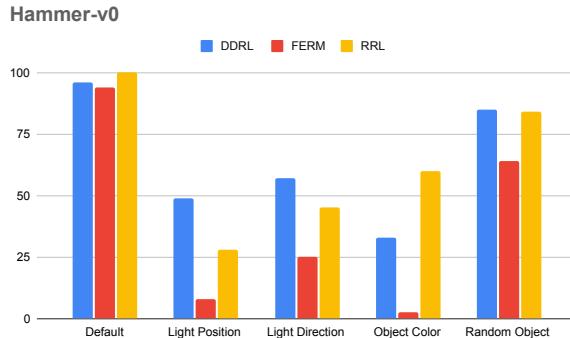


Fig. 3: Performance on visual distractors : In order to test the robustness of the policy, we expose the environment to various type of visual distractors in the scene like change in lightning conditions, random object, change in color etc. We observe that DDRL outperforms FERM in all the evaluations and competitive performance to RRL while being more robust to lightning conditions.

Furthermore, in Figure 3 similar to RRL [43], we subject the policy to various visual distractors during evaluation and observe superior performance to the competing baselines in most scenarios like lightning condition, presence of an random object into the scene. Interestingly, we observe a degradation in performance when subjected to change in color during inference, this suggests that the descriptors learnt must be exposed to a broader set of object instances (only one in this experiment) in order to make it more robust to the color changes in the object.

B. Generalization performance

The goal is to understand a) how well does the representation learned generalises to new unseen objects. b) is it possible to learn a policy that uses this representation and

completes the task with novel/unseen objects which require similar semantic understanding.

Qualitative analysis of the dense descriptor

For testing the performance of the dense descriptor, we train the model on five tools that have a similar semantic understanding (handle, neck, head) – Kinfe, Screw Driver, Spatula, Ratchet, Turner [Refer to Figure 4 in BLUE]. During evaluation, we expose the model to a completely novel object – Hammer [Refer to Figure 4 in RED]. We observe that the model really builds an understanding of different parts of the tools for e.g, neck is in the whitish pink space, the handle is in the pink space (in the visualization) in all the tools seen/unseen. Having such a generalised representation can help learn a better, sample efficient policy that pays attention to the right part of the tool while avoiding overfitting.

Policy Learning

In order to understand how good the policy generalizes to novel objects when the representations acquired using seen objects, we define the task of grasping and relocating the tool to a target position and use behavior cloning to learn the policy. The policy learned using 50 demonstrations for behavior cloning, we achieve a success rate of **47% on seen objects** and **65% on unseen object**. This provides some rudimentary evidence that learning a representation with a good semantic understanding helps in better generalization of the policy to novel objects.

V. LIMITATIONS

While this work demonstrates the improvement in generalization performance in policy learning, it still requires extra samples from the environment to learn the dense descriptor beforehand. Although in our experiments, we collect reasonable 200 trajectories, this might scale with the increasing complexity of the tasks and get more difficult to acquire. Furthermore, the dense descriptor adds an additional compute intensive layer when compared to RRL that directly uses Resnet-34 features.

VI. CONCLUSION

DDRL learns a semantically meaningful dense descriptor that is robust to the changes in the surrounding. This dense descriptor provides a good representation of the noisy image acquired from on board cameras, which in turn helps in improving the generalisation performance of the agent. Such representations acquired also help in improving the sample efficiency of the agent since the descriptor is free of noise and irrelevant information. We benchmark the sample efficiency of DDRL on the ADROIT dexterous manipulation benchmark and show robustness of the policy learned to various types of visual distractors. Finally, provide some elementary evidence that the descriptors learned can be used to generalize across different tools and even to previously unseen objects.

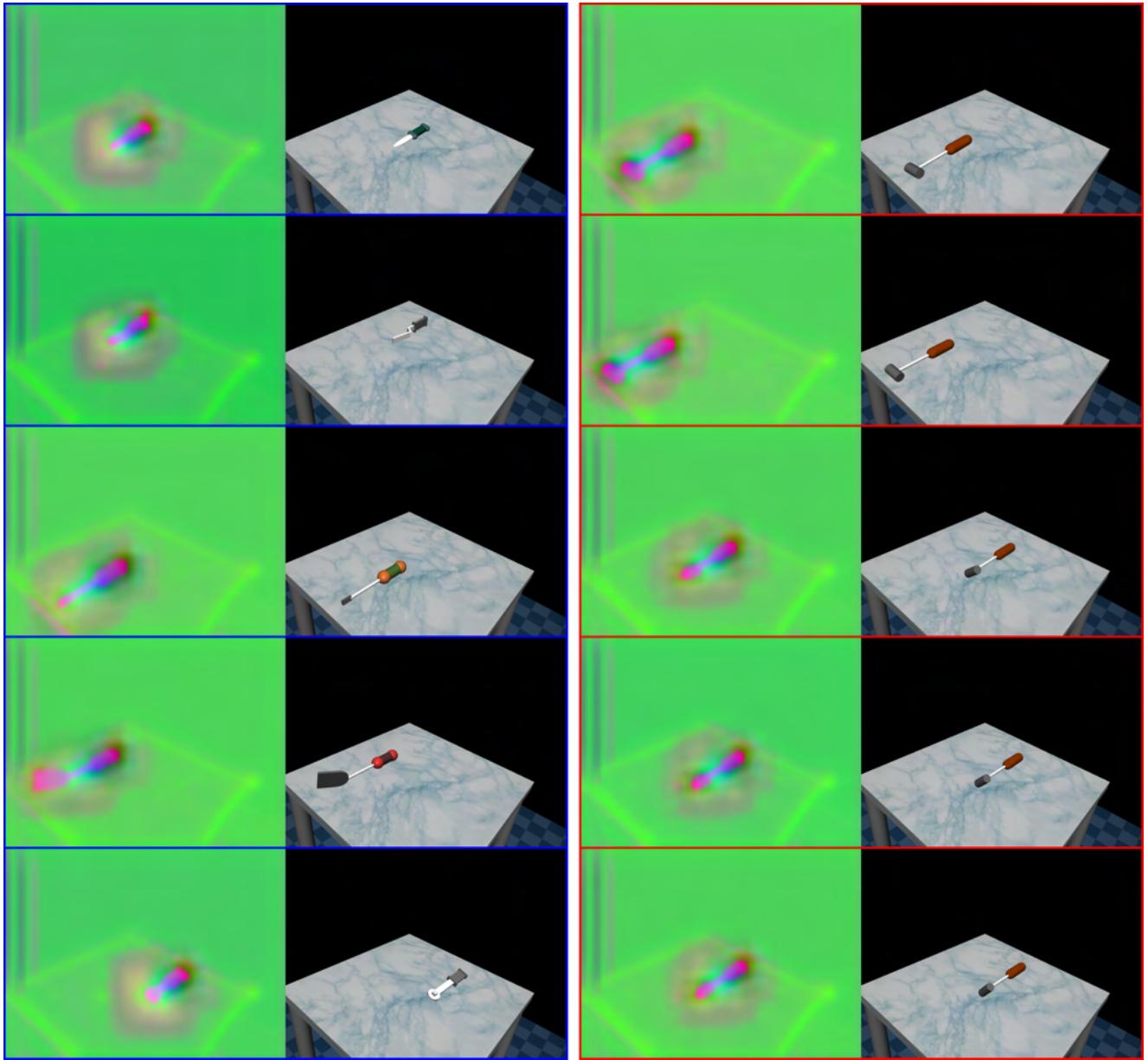


Fig. 4: Visualisation of the descriptor space. LEFT : Set of tools used for training the dense descriptor. RIGHT : During test time, the descriptor is exposed to a completely unseen object. We observe that the descriptor remains consistent across novel objects and provides a good semantically meaningful representation.

REFERENCES

- [1] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, “Mastering chess and shogi by self-play with a general reinforcement learning algorithm,” *arXiv preprint arXiv:1712.01815*, 2017.
- [2] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing atari with deep reinforcement learning,” *arXiv preprint arXiv:1312.5602*, 2013.
- [3] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, *et al.*, “Dota 2 with large scale deep reinforcement learning,” *arXiv preprint arXiv:1912.06680*, 2019.
- [4] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. V. D. Driessche, T. Graepel, and D. Hassabis, “Mastering the game of go without human knowledge,” *Nature*, vol. 550, pp. 354–359, 2017.
- [5] Z. Jiang, D. Xu, and J. Liang, “A deep reinforcement learning framework for the financial portfolio management problem,” *arXiv preprint arXiv:1706.10059*, 2017.
- [6] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, “Deep direct reinforcement learning for financial signal representation and trading,” *IEEE transactions on neural networks and learning systems*, vol. 28, no. 3, pp. 653–664, 2016.
- [7] C. Yu, J. Liu, and S. Nemat, “Reinforcement learning in healthcare: A survey,” *arXiv preprint arXiv:1908.08796*, 2019.
- [8] O. Gottesman, F. Johansson, M. Komorowski, A. Faisal, D. Sontag, F. Doshi-Velez, and L. A. Celi, “Guidelines for reinforcement learning in healthcare,” *Nature medicine*, vol. 25, no. 1, pp. 16–18, 2019.
- [9] J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.
- [10] J. Peters, S. Vijayakumar, and S. Schaal, “Reinforcement learning for humanoid robotics,” in *Proceedings of the third IEEE-RAS international conference on humanoid robots*, 2003, pp. 1–20.
- [11] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in *International conference on machine learning*

- learning.* PMLR, 2015, pp. 1889–1897.
- [12] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [13] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, “Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations,” in *Proceedings of Robotics: Science and Systems (RSS)*, 2018.
- [14] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al., “Solving rubik’s cube with a robot hand,” *arXiv preprint arXiv:1910.07113*, 2019.
- [15] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al., “Q-opt: Scalable deep reinforcement learning for vision-based robotic manipulation,” *arXiv preprint arXiv:1806.10293*, 2018.
- [16] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, et al., “Soft actor-critic algorithms and applications,” *arXiv preprint arXiv:1812.05905*, 2018.
- [17] H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, and S. Levine, “The ingredients of real-world robotic reinforcement learning,” *arXiv preprint arXiv:2004.12570*, 2020.
- [18] T. Verguts and W. Fias, “Representation of number in animals and humans: A neural model,” *Journal of cognitive neuroscience*, vol. 16, no. 9, pp. 1493–1504, 2004.
- [19] M. Laskin, A. Srinivas, and P. Abbeel, “CURL: Contrastive unsupervised representations for reinforcement learning,” in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, H. D. III and A. Singh, Eds., vol. 119. PMLR, 13–18 Jul 2020, pp. 5639–5650. [Online]. Available: <http://proceedings.mlr.press/v119/laskin20a.html>
- [20] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, “Reinforcement learning with augmented data,” *arXiv preprint arXiv:2004.14990*, 2020.
- [21] R. Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried, “Invariant visual representation by single neurons in the human brain,” *Nature*, vol. 435, no. 7045, pp. 1102–1107, 2005.
- [22] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, “kpam: Keypoint affordances for category-level robotic manipulation,” *arXiv preprint arXiv:1903.06684*, 2019.
- [23] A. Stone, O. Ramirez, K. Konolige, and R. Jonschkowski, “The distracting control suite—a challenging benchmark for reinforcement learning from pixels,” *arXiv preprint arXiv:2101.02722*, 2021.
- [24] N. Hansen and X. Wang, “Generalization in reinforcement learning by soft data augmentation,” *arXiv preprint arXiv:2011.13389*, 2020.
- [25] S. S. Du, S. M. Kakade, R. Wang, and L. F. Yang, “Is a good representation sufficient for sample efficient reinforcement learning?” *arXiv preprint arXiv:1910.03016*, 2019.
- [26] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.
- [27] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018.
- [28] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor,” in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.
- [29] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese, “Keto: Learning keypoint representations for tool manipulation,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 7278–7285.
- [30] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar, “Deep dynamics models for learning dexterous manipulation,” in *Conference on Robot Learning*. PMLR, 2020, pp. 1101–1112.
- [31] B. Chen, P. Abbeel, and D. Pathak, “Unsupervised learning of visual 3d keypoints for control,” *arXiv preprint arXiv:2106.07643*, 2021.
- [32] P. Florence, L. Manuelli, and R. Tedrake, “Self-supervised correspondence in visuomotor policy learning,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 492–499, 2020.
- [33] L. Manuelli, Y. Li, P. Florence, and R. Tedrake, “Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning,” *arXiv preprint arXiv:2009.05085*, 2020.
- [34] C. Finn, X. Y. Tan, Y. Duan, T. Darrell, S. Levine, and P. Abbeel, “Deep spatial autoencoders for visuomotor learning,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 512–519.
- [35] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, “Dream to control: Learning behaviors by latent imagination,” *arXiv preprint arXiv:1912.01603*, 2019.
- [36] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus, “Improving sample efficiency in model-free reinforcement learning from images,” *arXiv preprint arXiv:1910.01741*, 2019.
- [37] D. Dwibedi, J. Tompson, C. Lynch, and P. Sermanet, “Learning actionable representations from visual observations,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1577–1584.
- [38] I. Kostrikov, D. Yarats, and R. Fergus, “Image augmentation is all you need: Regularizing deep reinforcement learning from pixels,” *arXiv preprint arXiv:2004.13649*, 2020.
- [39] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq, et al., “Deepmind control suite,” *arXiv preprint arXiv:1801.00690*, 2018.
- [40] P. R. Florence, L. Manuelli, and R. Tedrake, “Dense object nets: Learning dense visual object descriptors by and for robotic manipulation,” *arXiv preprint arXiv:1806.08756*, 2018.
- [41] T. Schmidt, R. Newcombe, and D. Fox, “Self-supervised visual descriptor learning for dense correspondence,” *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 420–427, 2017.
- [42] R. Finman, T. Whelan, M. Kaess, and J. Leonard, “Toward lifelong object segmentation from change detection in dense rgb-d maps,” *2013 European Conference on Mobile Robots*, pp. 178–185, 2013.
- [43] R. Shah and V. Kumar, “Rrl: Resnet as representation for reinforcement learning,” in *International Conference on Machine Learning*. PMLR, 2021.
- [44] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [45] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al., “Imagenet large scale visual recognition challenge,” *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [46] S. Kakade, “A natural policy gradient,” in *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, ser. NIPS’01. Cambridge, MA, USA: MIT Press, 2001, p. 1531–1538.
- [47] A. Zhan, P. Zhao, L. Pinto, P. Abbeel, and M. Laskin, “A framework for efficient robotic manipulation,” *arXiv preprint arXiv:2012.07975*, 2020.
- [48] A. Rajeswaran, K. Lowrey, E. Todorov, and S. Kakade, “Towards Generalization and Simplicity in Continuous Control,” in *NIPS*, 2017.
- [49] S. Levine and V. Koltun, “Guided policy search,” in *Proceedings of the 30th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, S. Dasgupta and D. McAllester, Eds., vol. 28, no. 3. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 1–9. [Online]. Available: <http://proceedings.mlr.press/v28/levine13.html>
- [50] T. Kulkarni, A. Gupta, C. Ionescu, S. Borgeaud, M. Reynolds, A. Zisserman, and V. Mnih, “Unsupervised learning of object keypoints for perception and control,” *arXiv preprint arXiv:1906.11883*, 2019.

Discovering Affordances from Visual Perception in the Carnegie Mellon Ballbot

Dimitria Silveria¹, Oliver Kroemer² and Saumya Saxena²

Abstract—In order to interact with the environment efficiently, autonomous manipulators need to learn objects’ affordances. An effective way to do that is to allow the robot to build up its own experiences over time, similarly to infants’ motor skills development. In this paper, this infant-like developmental framework was applied in the Carnegie Mellon Ballbot learning process, by means of simulation. Force and vision sensors feedback were combined with the clustering algorithm X-means to detect affordances. After that, a visual classifier was trained so the robot could predict which affordances corresponded to each object, based on vision sensor feedback.

Index Terms—Affordances, Robotics Manipulators, Developmental Framework, Visual Perception, Ballbot

I. INTRODUCTION

The Carnegie Mellon ballbot is an autonomous mobile manipulator equipped with two 7-DoF arms [1]. It is designed to assist humans in tasks such as pushing into a wheelchair while balancing itself over a ball. In order to execute its manipulation tasks efficiently, the robot needs to learn the objects’ affordances.

Affordances are all the possible interactions that an environment provides to an agent [2]. For example, a wheelchair, with a human sitting in, can be pushed but it cannot be lifted up. The robot needs to have this kind of understanding of its action capabilities to assist humans effectively. As humans are excellent in perceiving such environmental characteristics [3], techniques based on infant development are effective ways to provide robot manipulators affordances of objects. [4] [5]. Therefore, they were chosen as the basis for this work.

In this research, the ballbot was provided with behavior primitives (grasp and push) in order to explore and learn these affordances through interaction, in a simulation environment. This way, the robot will be able to infer from its visual perception which interactions an object affords, preventing it to perform useless actions, like trying to lift a heavy table.

The learning process occurred in two phases [5]. The first stage corresponded to the unsupervised learning stage. The robot applied the grasp and push actions to different objects in the environment. Each of these interactions returned the forces exerted by the joints the difference between initial

¹Dimitria Silveria is with Electrical Engineering, Mechanics and Computer School, Federal University of Goias, Goiânia, GO 74690-900, Brazil dimitriasilveria.ds@gmail.com

²Oliver Kroemer is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA okroemer@andrew.cmu.edu

²Saumya Saxena is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA saumyas@andrew.cmu.edu

and final objects’ and actuator’s positions. We use these parameters to cluster these actions. Each cluster corresponds to how an object responded to an action.

In the second stage, the robot used RGB images patches, centered on each interaction point, as well as the clusters as labels to these images, to train a classifier.

Then, images from a test set were used as input to this classifier, so it could classify to each cluster each object belonged. Figure 1 represents an overview of the learning process.

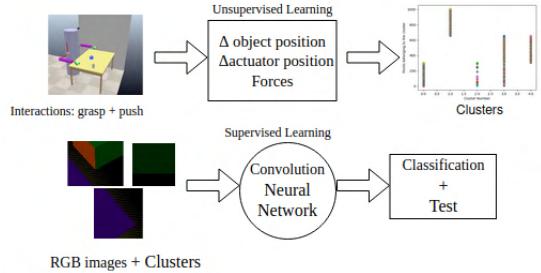


Fig. 1. Illustration of the learning process used in this work

II. RELATED WORK

Emre et al. [5] tested a three-staged developmental framework in an anthropomorphic manipulator. The manipulator was equipped with vision and tactile sensors. In Stage I, the robot executed random interactions with an object while collected the generated tactile effects. Some objects’ characteristics (size, shape, and position) were also collected by means of a vision sensor. These tactile signatures are clustered. Each cluster corresponded to a behavior primitive. At the end of this stage, the robot learned parameters of behavior primitives along with its tactile profiles. In Stage II, the robot executed the learned behaviors in different objects in order to conclude what behaviors were afforded by each object. After that, the robot learned to predict what effects each behavior caused on objects. In Stage III, the robot learned to perform complex tasks by imitating human tutors’ actions. The tutor performs their actions stopping at some points. The robot mapped these stopping points as subgoals and apply its previous knowledge to achieve these subgoals as well as the final goal. In our work, we used this idea of a learning process composed of an unsupervised stage followed by a supervised stage.

Savastano et al. [4] presented a sophisticated developmental framework, based on infants’ development. They designed

a neural controller and a sensory-motor system for the iCub humanoid robot. The neural controller was composed by a recurrent neural network that received input from many sensors along the robot's superior body. The sensory-motor system was composed by two connected layers in this neural network: a sensory and a motor. These two systems combined emulated the neural and motor development of infants in their first 4 to 18 months, by means of adjusting the neurons gains according to each phase. The robot undergoes a trial and error learning process, divided in three phases. In the pre-reaching phase the robot was equipped with head orientation and grasping reflex behaviors and a low vision insight. The distal DoF was frozen and the robot used motor babbling to perform the interactions in this phase. In the gross-reaching phase, the robot had its visual skills improved. The distal DoF was un-freezed and the use of motor babbling was reduced. In the fine-reaching state, the role played by the vision increased. With more exploration of the target and hand visual information, the robot improved its reaching and grasping reflexes. We didn't reproduce the exact idea of this paper, but we used the fine-reaching state as inspiration for our learning process. The ballbot executed the environment exploration counting on more refined reaching and grasping reflexes and visual perception. (To see more details, read section IV-B)

The work in [6], consists of collecting data from many sensors employing interactive exploration to train a manipulator to identify different food characteristics even without having previous contact with that kind of food before. The robot interacted autonomously with many foods and data were acquired from vision, audio, proprioceptive, and force sensors. First, they trained a neural network to output the food embeddings. Then they used these food embeddings to train perceptron classifiers and regressors. After that, the trained classifier was used in a food class that they had left out of the classification. The classifier was supposed to determine food type, hardness, juiciness slice type, and to predict slice width. Similarly, our research also trained a visual classifier, in order to provide the ballbot with the ability to determine affordances of objects from visual perception.

Gao et al. [7] proposed a model that took as input the semantic point and a dense geometry to plan robot trajectories to accomplish tasks even in the objects that the robot had never seen before. These 2 representations combined allowed the robot to infer about object physical characteristics, such as collision, static equilibrium and etc., and also extend these conclusion to same category objects that it had never seen before. To detect the 3D points, the authors used the kPAM method, which consists of representing the object by a set of 3D key points. Each robot action was represented by a rigid transformation applied to one of these points. To obtain the dense geometry, the ShapeHD method was used, to turn the RGBD images into 3D points. Based on this work, we used dense geometry and 3D points to plan robot trajectories.

The work developed by [8] consists of analyzing depth or color images from objects to infer which action primitives

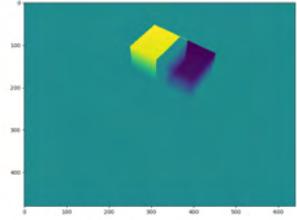


Fig. 2. Difference between initial and final depth images

are afforded by a pixel/point. The robot, in a simulation, was given six behavior primitives to interact with points selected, randomly, from the images. They took the ground truth position of the object, to narrow down the region of action from the point clouds. Each interaction could be classified as successful or not, according to a scoring system. In our work, we used a similar strategy to select the point of action, but instead of six interaction primitives, our robot was provided with two.

III. TECHNICAL APPROACH

A. Data Collection

The two interaction primitives (see more details in section IV-B) were realized in three frames, in order to analyze affordances from different frames. In the global and robot frames, the grasp and push actions performed in the +x, -x, +y, -y, +z and -z directions. In the object frame, the objects were grasped and pushed in the normal and tangent directions. The table and the cuboid were also pushed downwards.

First, each interaction primitive returned an output vector with 9 parameters:

$$\phi = [f_x, f_y, f_z, \Delta_x^{obj}, \Delta_y^{obj}, \Delta_z^{obj}, \Delta_x^{act}, \Delta_y^{act}, \Delta_z^{act}] \quad (1)$$

where f_x, f_y, f_z are the forces, collected directly from each joint, using vrep built-in function. We tried to collect data from the force sensors, at the fist moment, but we realized that collecting forces directly from the joints was a better approach for our work. $\Delta_x^{obj}, \Delta_y^{obj}, \Delta_z^{obj}$ are delta, the difference between the initial and final object positions, read from the respective point clouds. To calculate delta, the depth images before and after the interaction were subtracted and two point clouds were generated from the result. One for the negative points and other for the positive points. Then the average of these two point clouds were calculated and subtracted. The negative points corresponded to the points that were foreground before the interaction and became background after that. The positive points were the opposite. Figure 2 shows the difference between two depth images before and after one interaction.

$\Delta_x^{act}, \Delta_y^{act}, \Delta_z^{act}$ are the actuator position, which was the difference between the initial and final actuator positions, provided by v-rep. These parameters were collected in the global, robot and object frames.

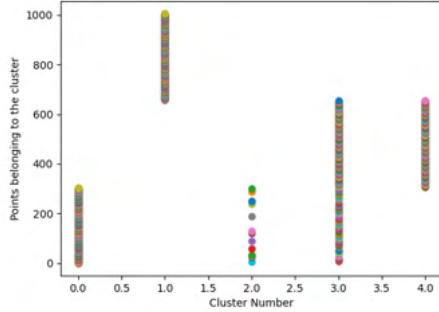


Fig. 3. Clusters in the global frame (x axis) and the points belonging to them (y axis)

B. Clustering Effects

The algorithm X-means [9] was chosen to cluster the data because it is able to decide how many clusters are necessary. The feature vector was the vector showed in the equation 1, for each interaction.

First, all the data from the 3 frames we clustered, together. However, it wasn't possible to take any significant conclusion from the results, because there were many clusters.

Then, we decided to cluster the data from the 3 different frames separately.

Each cluster was composed of points, on the objects, that provided similar outputs for one action. This means that they could represent actions that the objects affords or not (more details in section VI)

C. Clustering Variations

1) *Global Frame*: In figure 3 it is possible to see the 5 clusters obtained, in the global frame, and all the points belonging to them.

The points up to 100 correspond to points in the cuboid. The points from 101 to 650 were on the table and the rest were on the wall. There are 2 clusters containing only cuboid points, one for the wall points, 1 for table points, and one containing table and cuboid points.

The two clusters with cuboid points corresponded to the grasp and push affordances. There was only one cluster with wall points representing the fact that the wall doesn't have any affordance. There was a cluster with points in the table and the cuboid because whenever the robot tried to push the cuboid down, it didn't move. In this sense, both, the table and the cuboid, don't have this affordance. The cluster containing only table points represents the grasp action, which the table doesn't afford.

2) *Object Frame*: In figure 4 it is possible to see the 5 clusters obtained, in the object frame, and all the points belonging to them.

Cluster 0 contained only table points representing the grasp and push tangent direction affordances for this object. Cluster number 1 represented the outputs, from the push action in the normal direction, from the wall, and some of these from the cuboid. The cluster containing points on the

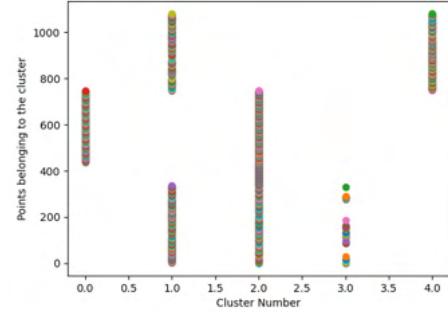


Fig. 4. Clusters in the object frame (x axis) and the points belonging to them (y axis)

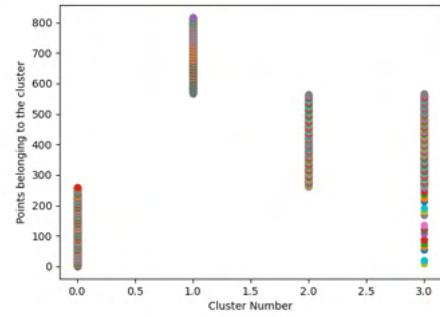


Fig. 5. Clusters in the robot frame (x axis) and the points belonging to them (y axis)

cuboid and on the table represented, the push downwards and in the normal direction affordances. Cluster number 3 represented all the grasp affordances plus push in a tangent direction, from the cuboid.

3) *Robot Frame*: In figure 5 it is possible to see the 4 clusters obtained, in the Robot frame, and all the points belonging to them.

Cluster number 0 grouped most of the cuboid affordances, except for the push downwards, that was represented in cluster number 3, together with the same affordance for points on the table. Cluster number 1 showed the fact that the wall doesn't afford any action. Cluster number 2, containing just table, points represent the grasp action, which the table also doesn't afford.

D. Classifying Visual Interaction Points

The RGB images from each interaction were cropped [10] in squares around the interaction point and used, as well as its corresponding cluster number to train a visual classifier [6] using the python library pytorch [11], for each reference frame. Figure 6 shows some examples of image patches used in the classifier.

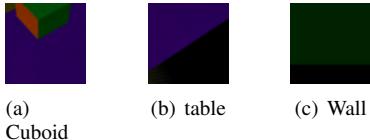


Fig. 6. Some image patches used in the classification

The classifier was trained using 85% of the data set. The other 15% were used as the test set

IV. EXPERIMENTS

A. Simulation Environment

This work didn't require a sophisticated model to represent the robot. Therefore we used a simplified one. The simulated robot was composed of a cylinder of 1.71m height and 0.5m diameter, without the ball in its base, because it was not necessary to include the robot dynamics in this work. Instead of the 7 DoF arms, the used robot had Cartesian arms, with 3 prismatic joints (each one moved in one ax of the Cartesian space), and a sphere playing the actuator's role. This arms' simplification was used because the focus was on the interactions with the object, regardless of the end-effector position in space. Figure 7 shows the difference between the two robots.

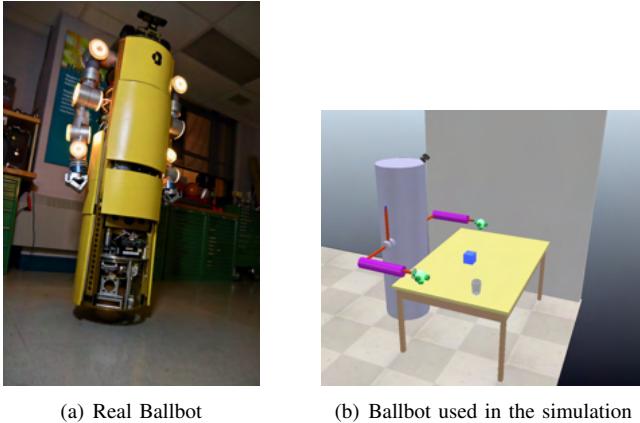


Fig. 7. The real ballbot and the ballbot used for this work

A force sensor was placed between the y joint and the final actuator, to measure the forces in each interaction. A Kinect sensor was placed on the cylinder's top, in order to collect depth and RGB images. There were also 4 force sensors attached to the final actuator, in order to emulate the grasp actions (this is detailed in section IV-B).

B. Interaction Primitives

Similarly to [12], the robot was given the reach, grasp, and additionally, push primitives. The actions were executed in three frames: global frame, robot frame, and object frame. In each frame, the robot performed the actions, in many directions.

The grasp reflex was emulated by combining the object's 3D coordinates with dense geometry information [7]. RGB and depth images were used with the library Open3D to

obtain the correspondent point cloud from the depth images. The 3D coordinates were used to extract only points on the object from the 3D representation. Then the robot selected random points in the object point cloud to interact with. The grasp movement was emulated by attaching the object to one of the 4 sensors in the final actuator. The push action was accomplished by attaching the sphere, sweeping the object.

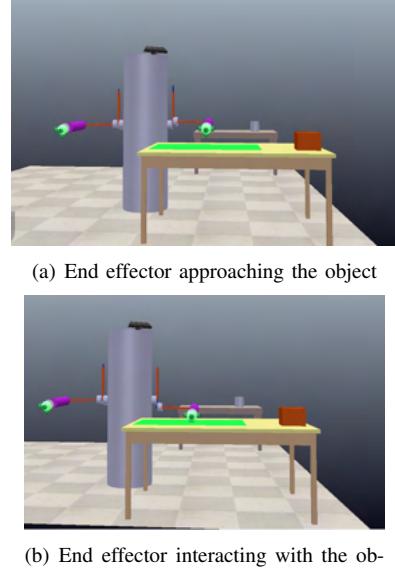


Fig. 8. The two steps in the grasp primitive, z direction

The grasp behavior was composed of three steps. In step one, the robot approached the object in the respective axis of action. In the second step, it grasped the object. In the third step, the actuator moved in the direction of the axis of action and upwards. After that, the object was released. The push behavior was executed in two steps. First, the robot approached the object in the respective axis of action, then, it swept the object, linearly, in the direction of the respective axis action.

Figure 8 shows two steps for the grasp action in the z-direction. In the sub-figure a), the robot is approaching in the z-direction. In the sub-figure b), it is trying to grasp the table.

V. RESULTS

A. Global Frame

The classifier showed an 82% accuracy over the test set.

Cluster	Accuracy
0	100%
1	100%
2	0%
3	0%
4	100%

TABLE I
EACH CLUSTER'S ACCURACY OVER THE TEST SET, IN THE GLOBAL FRAME

Table I shows each cluster performance over the test data set. Clusters number 0, 1 and 4, had a 100% accuracy, while the others had 0%

B. Object Frame

The classifier predicted correctly 52% of the images in the test set.

Cluster	Accuracy
0	0%
1	0%
2	100%
3	0%
4	100%

TABLE II

EACH CLUSTER'S ACCURACY OVER THE TEST SET, IN THE OBJECT FRAME

Table III shows each cluster performance over the test set. Clusters 2 and 3 had 100% accuracy, while the others had 0%.

C. Robot Frame

The classification algorithm predicted correctly 84% of the images from the test set.

Cluster	Accuracy
0	100%
1	100%
2	100%
3	0%

TABLE III

EACH CLUSTER'S ACCURACY OVER THE TEST SET, IN THE ROBOT FRAME

Table III shows each cluster performance over the test set. Clusters 0, 1, and 2 had 100% accuracy, while the number 3 had 0%.

VI. DISCUSSION

Comparing the clusters in the three frames, it is possible to realize that although the robot performed the same actions in the global and object frame, it got different numbers and kinds of clusters.

The classifiers presented good results for the 3 frames. However, there were two issues preventing it from present better results. The first one is that the robot performed multiple interactions with the same point, but collected the same image for all of them. Therefore, similar cuboid images, for example, could belong to the grasp and push downwards clusters, which makes it hard for the classifier to distinguish them. Another issue was a delay in the vrep Kinect sensor, which made it collect the images as though the object was in the final position of the last interaction. It didn't harm the table and wall pictures, after all, they didn't move. However, it affected the cuboid images, capturing only the table, in some of them.

VII. CONCLUSION AND FUTURE WORK

The proposed method was able to autonomously detect clusters corresponding to different objects and interaction constraints. This means that the framework developed is efficient to detect affordances from visual perception.

It is possible to conclude that the reference frames influence affordances perception.

In future work, it is necessary to collect data, with different objects, and test the classifier in objects different from those used for training it. It is also important to implement a classifier capable of returning more than one label per image. It is also crucial to use a better vision sensor, to avoid delay in the images capturing. It is also fundamental to implement this framework in the real robot

REFERENCES

- [1] R. Shut and R. Hollis, "Development of a humanoid dual arm system for a single spherical wheeled balancing mobile robot," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 499–504.
- [2] J. Greeno, "Gibson's affordances," *Psychological review*, vol. 101, pp. 336–42, 05 1994.
- [3] K. Khetarpal, Z. Ahmed, G. Comanici, D. Abel, and D. Precup, "What can I do here? A theory of affordances in reinforcement learning," *CoRR*, vol. abs/2006.15085, 2020. [Online]. Available: <https://arxiv.org/abs/2006.15085>
- [4] P. Savastano and S. Nolfi, "A robotic model of reaching and grasping development," *IEEE Transactions on Autonomous Mental Development*, vol. 5, no. 4, pp. 326–336, 2013.
- [5] E. Ugur, Y. Nagai, E. Sahin, and E. Oztop, "Staged development of robot skills: Behavior formation, affordance learning and imitation with motionese," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 2, pp. 119–139, 2015.
- [6] A. Sawhney, S. Lee, K. Zhang, M. Veloso, and O. Kroemer, "Playing with food: Learning food item representations through interactive exploration," 2021.
- [7] W. Gao and R. Tedrake, "kpam-sc: Generalizable manipulation planning using keypoint affordance and shape completion," *CoRR*, vol. abs/1909.06980, 2019. [Online]. Available: <http://arxiv.org/abs/1909.06980>
- [8] K. Mo, L. J. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani, "Where2act: From pixels to actions for articulated 3d objects," *CoRR*, vol. abs/2101.02692, 2021. [Online]. Available: <https://arxiv.org/abs/2101.02692>
- [9] D. Pelleg and A. Moore, "X-means: Extending k-means with efficient estimation of the number of clusters," *Machine Learning*, p. 01 2002.
- [10] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," *CoRR*, vol. abs/1509.06825, 2015. [Online]. Available: <http://arxiv.org/abs/1509.06825>
- [11] P. Tutorial, "Training a classifier," https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html#training-a-classifier, August 2021.
- [12] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, Y. Yoshikawa, M. Ogino, and C. Yoshida, "Cognitive developmental robotics: A survey," *Autonomous Mental Development, IEEE Transactions on*, vol. 1, pp. 12 – 34, 06 2009.
- [13] Open3D.org, "Training a classifier," http://www.open3d.org/docs/release/python_api/open3d.geometry.PointCloud.html, July 2021.

ACKNOWLEDGEMENTS

I would like to thank Carnegie Mellon Robotics Institute and the Intelligent Autonomous Manipulation lab, for providing me with this opportunity. I also would like to thank everyone who, somehow, worked for this program to happen, especially Ms. Rachel and Dr. John Dolan. And last, but not least, I would Like to thank my mentors Dr. Oliver Kroemer and Saumya Saxena for being kind and helpful.

Explanations in Multi-Agent Search and Rescue Task

Manav Singhal¹, Vidhi Jain², Dana Hughes², and Katia Sycara²

Abstract—Interpreting the reasoning behind decisions being taken by an agent is as important as making the right decision. Once the reasoning behind the decision is understood, the decision making process can also be improved by making timely interventions. In a search and rescue task, the importance of each decision is very high as the life of a human is at stake. Currently very few works have delved into interpreting actions. Prior to our study, in the search and rescue task belief modeling has been experimented in the single-agent setting to comprehend the belief of the agent about its environment while making its decision. Given that these tasks are generally carried out in a team which changes the dynamics of decision making, our study aims to model the belief that each agent possesses about other agents and objects at a given time-step influencing the decisions it takes.

Index Terms—belief modeling, multi-agent, search and rescue task

I. INTRODUCTION

The theory of mind is the human capacity for reasoning about agents' mental states such as beliefs and desires. It deals with explaining and predicting the observable actions taken by an agent at a given time-step based on the belief and desires held by the agent. The unobservable mental state depends on the information possessed by the agent about the environment around it. The inconsistencies in beliefs hence actions arise from differences between the past and the present reality, as well as the mental states of agents who may have false beliefs about the world or about the mental states of other agents.

A. Theory of Mind Experiments

The famous Sally-Anne experiment goes as follows [1]:

- Sally and Anne are two friends who have two containers, a basket and a box respectively.
- After putting a marble in her basket, Sally leaves the room (and is not able to observe the events anymore).
- After Sally's departure, Anne moves the marble to her box.
- Then, Sally returns to the room.

Based on the false belief possessed by Sally, she is expected to search the basket for the marble, however the reality of the situation is that the the marble exists in the box. This experiment helps in posing three questions evaluating the theory of mind for which the answers can be sought from the observers:

¹Manav Singhal is with the Department of Electrical and Electronics Engineering, National Institute of Technology Karnataka, Surathkal, India manavsinghal15@gmail.com

²Vidhi Jain, Dana Hughes, and Katia Sycara are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA {vidhij, danahugh, sycara}@andrew.cmu.edu

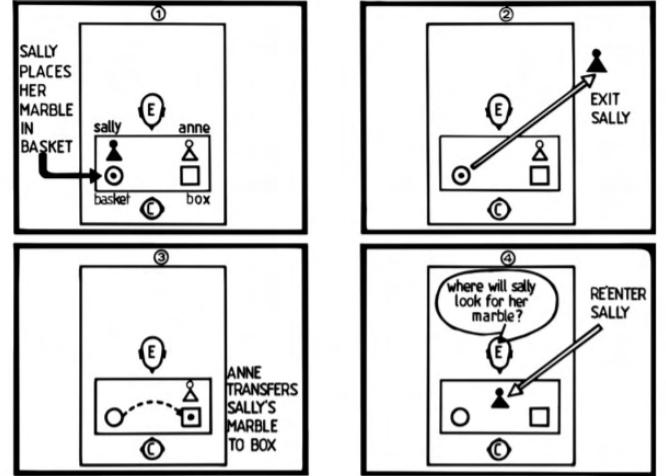


Fig. 1: The Sally-Anne experiment setup from [1]

- Where will Sally look for the marble? [First-order belief]
- Where was the marble at the beginning? [Memory]
- Where is the marble really? [Reality]

The first question test the reasoning ability about Sally's belief about the location of the marble. Interestingly [1] found that most children below the age of three were unable to answer this question correctly as Sally's belief is different from the reality of the world.

This experiment, however, only modeled the first order belief. People can even reason about other's beliefs: Anne believes that Sally believes that the marble is in the basket. [2] experimented on second order beliefs. In their setup Mary and John see an ice-cream van in the park, and the ice-cream man tells them that he will be in the park until later in the afternoon. Mary leaves the park and goes home. The ice-cream man then decides to leave the park informing John that he is going to the church. On the way, on seeing Mary he informs her that he will be selling ice-creams close to the church all afternoon. The question then posed was: "Where does John think Mary goes to get the ice-cream?". Thus, John has a second-order false belief about Mary's belief. This is a more complex cognitive task.

B. Related Work

To represent the agent desires and beliefs about the environment, [3] models the agent planning and inference problem as a Partial Observable Markov Decision Process (POMDP). From the agents behaviour in the environment, the beliefs can be attributed. A Dynamic Bayes Net (DBN)

is employed to jointly model belief and desire inference over time.

Inspired by the Sally-Anne experiment [1], and building on the Bayesian theory of mind [3], [4] creates a fresh theory of mind framework representing the mental state of minds. This framework determines the actions and beliefs from a latent state representation of the observation. It builds a strong prior model to adapt to richer predictions and mental states using a small number of observations thus increasing the generalization capability. The limited data autonomously modeling helps in greater reach of the framework. The agent uses meta-learning to develop the prior from the behaviour of the new agents.

Adopting visual cues to build on the theory of mind, [5] focuses on identifying which agent has an incorrect belief and when they are mistaken in a simple animated story. A person can have a false belief for reasons such as including occlusion or misinterpreting intentions. A person-centric approach has been adopted to understand the belief of the agent. A convolutional logistic regression is used for learning over the images in the animated story. To answer the question of who is mistaken, the classifier response is marginalized across time while to answer the question of when they are mistaken, the classifier response is marginalized across people.

In contrast to the visual cues [5], [6] took to the language modeling of the stories to model beliefs. [6] helps in comparing existing theory of mind models on three created datasets inferring beliefs from textual stories. Their proposed dataset helps in addressing the problems in the existing bAbI dataset [7] as they help to incorporate the observers of each event in the story as well as add on to task of finding answers to the second-order belief of the Sally-Anne experiment. Keeping track of the belief held by an agent about other agents is a hard problem. Hence, this resulted in existing models performing worse on the dataset in comparison to the one in [7].

None of the prior works have yet specifically delved into the nuances of the search and rescue task. Hence, in our work we aim to do that by using the insights from [6] to model our data generation for the search and rescue task as well as create our framework inspired by [8] by maintaining an observation annotation matrix.

Section II defines the general search-and-rescue problem and our initial experiment design. Section III describes the results of belief questions posed to the agent about the location of the victim. Section IV concludes the main takeaways from the results, brief about the challenges faced, and discusses the future work.

II. METHODOLOGY

In this section, we describe the actual experiment design for the search and rescue task performed by humans players in the Minecraft environment along with our current basic design to model the beliefs simulating a toy scenario of the



Fig. 2: Our current experiment map

case. We further delved into the data format and highlight our technique to model the beliefs of the agents.

A. Experiment Design

In the search and rescue task, human players are given the task of navigating an office building simulated in the Minecraft environment [9]. There are three players who in a collaborative effort try to rescue victims in this building within a stipulated time. The three roles include a medic who can triage the victims, a searcher who can move victims from one location to another, and an engineer who can break the rubble that might be found at different locations of the map due to the disaster scenario. The players are supposed to save both the critical victims yielding more reward and the regular victims yielding lesser reward with the goal of maximizing their score. The players are given partial maps indicating the locations of a few victims. They also have communication amongst the players to discuss their strategies. In addition to this, the players also have three different markers representing no victim, a regular victim or a critical victim which they can place to convey their observation of a particular location to other players. However, the marker semantics given to the three players varies, thus inducing false beliefs about the semantics amongst the players.

Inspired by this design we broke the problem down to a simpler version (Fig. 2), where we have three agents namely the engineer, searcher and medic having the same roles as mentioned above. With these three roles, we create a simple representation map with two rooms, the hall, and one victim in each room. We create stories illustrating a simple event of searcher moving a victim and then try to understand the belief possessed by the agent about the location of the victim.

B. Data Generation

Initially, we worked on manually creating the data to represent the dataset created in [6]. This includes stories in the form of sentences including the agent, its location, its activity, and the observers of this event.

Example Data	
Event	Observers
Medic enters Hall	1
Searcher enters Hall	1 2
Engineer enters Hall	1 2 3
Searcher enters Room2	1 2 3
Medic enters Room1	1 2 3
Engineer enters Room2	1 2 3
Searcher moves Victim2 from Room2 to Hall	2

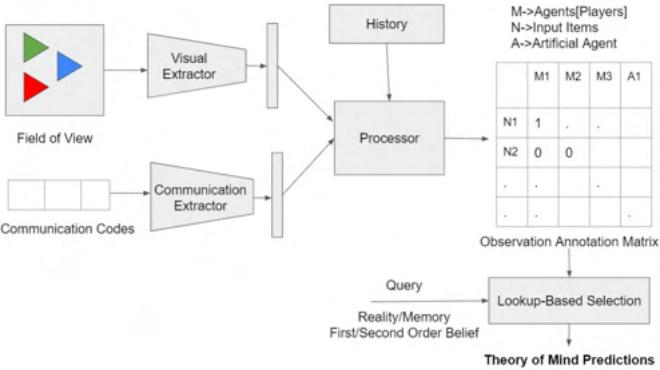


Fig. 3: The framework we aim to implement

The numbers in the observers section follow the following semantics: 1 is assigned to the medic, 2 is assigned to the searcher, and 3 is assigned to the engineer respectively.

Once we established the required format for our case, we worked on automating the data generation. We created key verbs such as 'picks', 'drops', 'triaxes', and 'enters'. From this list of verbs we randomly generated a verb and checked if it matched the activities that can be performed by the randomly generated role of the agent. The sequence of the verbs was kept in mind by generating verbs such as 'drops' only after verbs such as 'picks' so that the random generation does not lead to arbitrary data being generated. The observers for each line of input are generated from the matrix tracking the location of the agents.

C. Belief modeling

The memory network model (MemN2N) which was proposed by [10] used an external memory cache to read and write into the memory as the story is parsed. [8] builds on this work to incorporate an observation annotation matrix in addition to memory network to improve the performance. The observation annotation matrix is $(M + 1) * N$ matrix where $(M + 1)$ represents the number of agents along with the oracle agent while N represents the input items for a story. The matrix S is such that $S_{ij} = 1$ if input item x_i is observable to agent j and 0 otherwise, where they assign the oracle observer (who observes all input items) to the first index. [6] which compared the performance of the different models on the dataset created found the multiple observer model to perform best on the ToM-Easy (each story has one task) and second best in the ToM task with noise (stories with multiple tasks along with random events added in between). [8] demonstrates the cases of how an action can induce a belief or a belief can result in an action. We aim to see similar cases in our study. An example would be: An agent on seeing Marker1 enters the room indicating they believe a victim is there (Action inducing belief), An agent believes Marker1 indicates a regular victim is present hence enters the room (Belief leading to an action).

Given the good performance of [8], we aim to extend this model to our use case. In our model (Fig. 3), we aim to incorporate the agent observation along with the

Medic enters Hall. 1
Searcher enters Hall. 1 2
Engineer enters Hall. 1 2 3
Searcher enters Room1. 1 2 3
Medic enters Room2. 1 2 3
Engineer enters Room2. 1 2 3
Engineer enters Hall. 3
Searcher moves Victim1 from Room1 to Hall. 2 3

Fig. 4: An example story input describing the searcher moving the victim

Enter the question : Where does Medic believe Victim1 to be?
Medic believes Victim1 is in the Room1
Enter the question : Where was Victim1 at the beginning?
Victim1 was in the Room1 in the beginning
Enter the question : Where is the Victim1 actually?
Victim1 is actually in the Hall

Fig. 5: Output for the above story

communication between the agents at each time-step. The field of view extractor is a grid within a perimeter of the agent at a time-step from which the other objects/agents present are represented. This information is passed through a learning based processor (neural network), which along with the history of beliefs of the agents will predict an integer in the observer annotation matrix of an input item corresponding to an agent. This integer is prior mapped to a location of the input item or mapped to an indication to a specific semantic. The training data for the learning based processor will be obtained from the data generation step where a similar matrix is generated for obtaining the observers of an event. This matrix will hence indicate the belief of the agent about an input item. The last column being the oracle observer or in our case the artificial agent has the ground truth values. From this matrix, based on the question posed following a lookup based selection the theory of mind predictions can be sought to answer the question about the belief of the agent.

This paper explores the memory processing and the observer annotation matrix of the proposed framework; incorporating vision and communication events from the test-bed being envisioned as future work.

III. RESULTS

In this section, we describe the results of our current implementation of the framework proposed. We generated stories to model the different scenarios that might be encountered when the searcher moves a victim to another location or the different marker semantics possessed by the agents. From the events in the stories we generated the observation annotation matrix as described in the previous section. Questions querying the matrix asking the questions similar to the Sally-Anne experiment [1] outputted the correct results with

respect to the memory, reality and first order belief. As it

```

Medic enters Hall. 1
Searcher enters Hall. 1 2
Engineer enters Hall. 1 2 3
Searcher enters Room1. 1 2 3
Medic enters Room2. 1 2 3
Engineer enters Room2. 1 2 3
Searcher places Marker1 at Room1. 2
Searcher enters Hall. 2
Engineer sees Marker1. 3 1

```

Fig. 6: An example story input describing the marker being placed

```

Enter the question : What does Engineer believe Marker1 to be?
Engineer believes Marker1 means Regular Victim Here

Enter the question : What is the Marker1 actually?
Marker1 actually means No Victim Here

```

Fig. 7: Output for the above story

can be seen in the story (Fig. 4), the Medic witnesses the event of the Searcher moving Victim1 from Room1 to the Hall. Given the ground beliefs possessed by the three agents, it can be seen that the Medic still believes Victim1 to be in Room1 (False Belief). The memory based question indicates correctly (Fig. 5) that the Victim1 was present in the Room1 initially while the reality question posed indicates correctly the Victim1 is in the Hall. (Fig.7) demonstrates the answers to the queries for the beliefs about the marker semantics and the reality of the case. Overall, this method showed promise in modeling the history of the locations of the agents and victims, thus adequately modeling the beliefs.

IV. CONCLUSIONS

We found promising initial results on using the observation annotation matrix to model the beliefs of the agents and answer the queries about the theory of mind posed. The observation annotation matrix helped to keep track of the various beliefs held by the players while navigating the environment in a model search and rescue task. We have currently finished implementing the data generation, the observation annotation matrix and the lookup-based selection for the false belief induced by the searcher moving a victim. We aim to scale the map and generate the field of view from the story to incorporate that along with the communication so that we could switch to a learning based method for the processor. Further, we aim to delve into changing the input items in the matrix into facts to standardize the value of the matrix into binary.

ACKNOWLEDGMENT

This work was supported by the CMU Robotics Institute Summer Scholars (RISS) Program. Manav Singhal would also like to thank Dr. John M. Dolan and Ms. Rachel Burcin for their constant support. Special thanks to Dr. Joseph Campbell for his feedback.

REFERENCES

- [1] S. Baron-Cohen, A. Leslie, and U. Frith, "Does the autistic child have a theory of mind?" *Cognition*, vol. 21, pp. 37–46, 11 1985.
- [2] J. Perner and H. Wimmer, "'John thinks that Mary thinks that...'" attribution of second-order beliefs by 5- to 10-year-old children," *Journal of Experimental Child Psychology*, vol. 39, no. 3, pp. 437–471, 1985. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/0022096585900517>
- [3] C. L. Baker, R. Saxe, and J. Tenenbaum, "Bayesian theory of mind: Modeling joint belief-desire attribution," *Cognitive Science*, vol. 33, 2011.
- [4] N. Rabinowitz, F. Perbet, F. Song, C. Zhang, S. M. A. Eslami, and M. Botvinick, "Machine theory of mind," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, 10–15 Jul 2018, pp. 4218–4227. [Online]. Available: <http://proceedings.mlr.press/v80/rabinowitz18a.html>
- [5] B. Eysenbach, C. Vondrick, and A. Torralba, "Who is mistaken?" *CoRR*, vol. abs/1612.01175, 2016. [Online]. Available: <http://arxiv.org/abs/1612.01175>
- [6] A. Nematzadeh, K. Burns, E. Grant, A. Gopnik, and T. Griffiths, "Evaluating theory of mind in question answering," *ArXiv*, vol. abs/1808.09352, 2018.
- [7] J. Weston, A. Bordes, S. Chopra, and T. Mikolov, "Towards ai-complete question answering: A set of prerequisite toy tasks," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: <http://arxiv.org/abs/1502.05698>
- [8] E. Grant, A. Nematzadeh, and T. Griffiths, "How can memory-augmented neural networks pass a false-belief task?" *Cognitive Science*, 2017. [Online]. Available: https://cocosci.princeton.edu/papers/grant_etal_17_cogsci.pdf
- [9] L. Huang, J. Freeman, N. Cooke, M. Cohen, X. Yin, J. Clark, M. Wood, V. Buchanan, C. Carroll, F. Scholcover, A. Mudigonda, L. Thomas, A. Teo, M. Freiman, J. Colonna-Romano, L. Lapujade, and K. Tatapudi, "Using humans' theory of mind to study artificial social intelligence in minecraft search and rescue," (*to be submitted to the Journal of Cognitive Science*, 2021).
- [10] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, p. 2440–2448. [Online]. Available: <https://arxiv.org/abs/1503.08895>

Evaluation of AutoML Systems Using OpenML Datasets

Maya Sitaram¹, Jieshi Chen², and Artur Dubrawski²

Abstract—Building machine learning (ML) models for real-world data problems requires collaboration between domain experts and data scientists, with manual programming efforts in data cleaning, featurization, hyperparameter tuning and model selection. With the introduction of automated machine learning (AutoML), these processes could be handled automatically. AutoML enables domain experts and data scientists to focus on the underlying problem construction, to quickly prototype solutions to new problems, and to boost productivity. As part of Data Driven Discovery (D3M) program by Defense Advanced Research Projects Agency (DARPA), the Auton Lab at Carnegie Mellon University (CMU) has created an automated machine learning tool, AutoⁿML, which has outperformed other competing D3M AutoML frameworks in several periodic DARPA evaluations. In this paper, we evaluated the performance of our AutoⁿML system with comparison to three open source AutoML tools such as H₂O AutoML, Tree-Based Pipeline Optimization Tool (TPOT) and Auto-Sklearn by computing lift statistics and rank statistics under different time limit experiment settings. We found that our AutoⁿML did not always outperform the other AutoML systems with the “top” 1 pipeline from training data. For some datasets, an alternative “best” pipeline achieves highest performance on unseen test data, which implies potential area of improvement of our AutoⁿML given the characteristics of datasets.

Index Terms—AutoML, automated machine learning, model selection

I. INTRODUCTION

Machine learning models need to be individually tuned with proper parameters and the best feature sets from the data provided to improve the performance. This manual tuning and data-cleaning process, also referred to as “Combined Algorithm selection and Hyperparameter optimization (CASH)”, requires the help of data scientists who are familiar with ML algorithms and statistics [1]. As a result, ML model development limits the involvement of experts familiar with the problem domain, and is bottlenecked by the number of available data scientists. With the advent of Automated Machine learning (AutoML), the best ML model given a prediction task is found automatically, allowing the end-user to output predictions from unformatted data, without relying on a data scientist [1].

An ML pipeline is a sequence of primitive algorithms or modeling steps that are combined to transform the input space (x) into target values (y). Primitive algorithms are generally learnable functions or models for data-cleaning, featurization, hyperparameter tuning, model fitting, evaluation, etc.

¹Maya Sitaram is with the Department of Mechanical Engineering and Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA msitara1@jhu.edu

²Jieshi Chen and Artur Dubrawski, PhD, are with the Auton Lab, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA jieshic@andrew.cmu.edu, awd@cs.cmu.edu

An AutoML framework generates multiple pipelines which are validated against a portion of the training data and ranked by the scoring metric defined specific to the modeling task. Then the pipeline that ranks the first on training data will be applied to predict the unseen test data [2].

AutoⁿML, which constructs prediction pipelines from a set of primitive algorithms, is an open-source AutoML system developed for the DARPA D3M project and has consistently outperformed other AutoML systems in periodic DARPA D3M evaluations. However, our system has not yet been evaluated against non-D3M open source AutoML systems such as H₂O AutoML and Auto-Sklearn. The following experiments aim to evaluate the performance of AutoⁿML and selected non-D3M AutoML systems and uncover the advantages and limitations of our AutoⁿML for further improvement.

In this paper, we evaluated the prediction performance of the Auton Lab AutoML framework, AutoⁿML, with those of selected open source AutoML frameworks, including Auto-Sklearn, TPOT, and H₂O AutoML. Each AutoML varies in the complexity of the available algorithms in its search space, searching processes and resulted pipelines. This paper is not about the differences in implementations or pipeline searching process among AutoML systems. Instead, we perform a comparison of multiple AutoML systems based on prediction performance given the same data task and same training time. The datasets used for this evaluation are limited to classification tasks of the OpenML datasets.

II. RELATED WORK

Studies on evaluation of open source AutoML frameworks often use established AutoML benchmark datasets from OpenML and Kaggle. Gsjibers et. al proposed an open source AutoML Benchmark based on 39 OpenML datasets [3], [4]. Other studies have been curated for the specific evaluation from Kaggle competitions or highly-dimensional datasets that have been prepared into training sets of increasing size (10K, 100K, 1 million,... instances) [1], [3]–[5].

A notable framework evaluation by Zoller and Huber, uses 137 datasets from OpenML benchmarking suites to evaluate six publicly-available AutoML systems: TPOT, Hyperopt-Sklearn, Auto-Sklearn, Random Search, Auto Tune Models (ATM), and H₂O AutoML. To establish baseline performance measures, a simple Random Forest pipeline and a Dummy Classifier used to make random predictions are also run in tandem. The evaluation is run with 8-core processors and 30GB memory. During evaluation, 4-fold cross validation is performed per dataset, where 3 folds are used for training and the last fold is used to calculate

test performance. The AutoML systems are provided with binary and multiclass classification tasks, using datasets which contain 500,000 to 600,000 samples and less than 7,500 features. The AutoML systems are trained for a soft limit of an hour, and a “hard limit” of 1.25 hours (at which point the program is aborted). The AutoML systems are compared using metrics such as average accuracy across all data sets, normalized performance compared to baseline learners, and average learning-test overfit (the difference between performance on the train dataset and performance on the test dataset). As a result, TPOT outperformed the most frameworks averaged over all datasets [1]. Through this objective evaluation, TPOT appears as a worthy candidate for comparison against AutoⁿML.

Other evaluations have been published by the developers of AutoML systems themselves, and present the results of their AutoML system performance compared to others’. For instance, an evaluation published by developers of AutoGluon Tabular compares their AutoML against five other frameworks: AutoWEKA, Auto-Sklearn, TPOT, H₂O AutoML, and Google Cloud Platform (GCP) Tables. Similar to the evaluation by Zoller and Huber, the evaluation is run with 8-core processors and 32GB memory. Binary and multiclass classification problems and regression problems are provided to the AutoML systems from the Gsjibers Benchmarking Dataset and selective datasets from previous Kaggle competitions, and the AutoML systems are trained for 1 hour. However, the metrics for comparison include the rank of each framework (1-6) and loss [i.e. 1-Area Under the Curve (AUC)]. In addition, AutoGluon Tabular looks at each frameworks’ relative performance as compared to AutoGluon AutoML – such as the number and types of datasets in which the framework produced better or worse predictions. As a result, AutoGluon ranks among the top two frameworks on average is more robust with less system failures [3]. Though AutoGluon is not included as a candidate in our experiments, this evaluation presents a list of AutoML frameworks commonly used for framework evaluation. It also presents potential metrics for comparison in our evaluation.

Similarly, the framework evaluation by developers of H₂O AutoML, compares H₂O AutoML to AutoWEKA, Auto-Sklearn, and TPOT. The hardware protocols, benchmark datasets, and training time limits for each AutoML are consistent with the previous two evaluations mentioned above. The metrics for evaluation include log loss and AUC per dataset. As a result, H₂O performs favorably compared to most frameworks on a variety of data types and for large datasets [4]. This presents H₂O AutoML as a worthy candidate for comparison against AutoⁿML.

Developers of Automatic Gradient Boosting AutoML (“Auto-xgboost”). evaluate performance with comparison to AutoWEKA and Auto-Sklearn. The experimental parameters of this evaluation differ slightly as it is run with 28-core processors and 64GB memory and the AutoMLs are given a maximum runtime of 6h. Surprisingly, Auto-Sklearn outperforms Autoxgboost and AutoWEKA on a majority of

the tested datasets (i.e. 9 out of 16 datasets), even though the paper suggests that Auto-Sklearn has a larger “tuning space” and more than one learning algorithm, which could present a disadvantage compared to the single-learning algorithm used in Autoxgboost [5]. However, given the performance results of Auto-Sklearn and its repeated mention in other framework evaluations, Auto-Sklearn is selected for comparison against our AutoⁿML. Patterns in the experimental setup and comparison metrics of the evaluations mentioned above also influence the methods of our evaluation.

III. METHODS

Experiments are setup to answer the following research questions pertaining to the input space and performance of different AutoML systems:

- How significantly does AutoⁿML outperform others in general? To answer this question, we computed relative performance measures, or lift metrics, by normalizing the performance of AutoⁿML and other AutoML frameworks by that of the baseline/majority-vote model.
- How does the distribution of predictions resulting from our AutoⁿML differ from those of other systems?
- What types of data tasks do our AutoML system outperform other frameworks by the defined metrics of the problem?
- What datasets do AutoⁿML fail to beat the other three AutoML systems?

We evaluated our AutoⁿML and selected three AutoML systems, H₂O AutoML, TPOT and Auto-Sklearn on 76 OpenML datasets of binary classification tasks. Experiments were run on an 8-core Linux machine. A systematic evaluation framework was written to automate the experiments. First, the evaluation framework randomly split each dataset into training and test datasets. Then, it passes the identical train-test split to each AutoML framework as inputs. With the given inputs, pipeline search and fitting were executed under three training time limits, which are 1 minute, 10 minutes, and 20 minutes.

The AutoML systems search for the most optimal prediction pipeline from training data to optimize the AUC score as the ranking metric. For each AutoML, top 10 pipelines and corresponding results were output and stored. For each pipeline, the output included a .json file of the pipeline’s model structure, the pipeline’s predictions of the training data, the pipeline’s predictions of the test data, and the pipeline’s AUC score on the training data.

The prediction files were used to calculate performance metrics during runtime as well. We used the predicted labels on the training data to calculate each pipeline’s accuracy score as a second performance metric. Similarly, we used the predicted labels on the test data to calculate each pipeline’s accuracy and AUC scores on the test data. To obtain a baseline accuracy score, a default classifier, which predicts the majority-class label from the dataset for every data point and performs similarly to a random classifier, was also computed and recorded. Those forementioned performance

metrics were collected in a summary .csv file for each (1-, 10-, and 20-minute) training time limit

The 76 OpenML datasets vary in data sizes as well as dimensionalities. Some high dimensionality and sparse datasets were chosen to challenge the AutoML systems' predictive abilities. For complex datasets, AutoⁿML would oftentimes override the allotted training time limit. Therefore, we allow all AutoML systems to exceed the training time limit if necessary, and the system execution will not be stopped.

We computed Rank and Lift metrics to normalize the performance across datasets for fair comparison and feasibility in summary statistics. For each dataset, the AutoML systems were ranked from 1 to 5 based on the AUC score of their predictions on the training data ("Training" AUC score). They were also ranked based on the AUC score of their predictions on the test data ("Test" AUC score). The "Test" Lift, or the relative performance of each AutoML compared to the default classifier, was computed by dividing its accuracy score by the default accuracy on the test dataset. Similarly, the "Training" Lift metric was also computed by dividing each AutoML's accuracy score by the default accuracy score on the training dataset.

IV. RESULTS

After reviewing the output predictions from AutoⁿML, in the majority of the 76 datasets evaluated, the "top"-scoring pipelines found from training data does not perform well on the unseen test data. As mentioned previously, the "top" pipeline has the highest AUC score on the training data amongst all the pipelines searched. For these datasets, however, the pipelines searched have poor predictive performance; their AUC scores on the training data fall below 0.1, within the possible range of 0 to 1. Thus, we suspect the pipelines might not be ranked properly by AUC, and the "top" pipeline outperforms the rest pipelines by a small margin. As we cannot guarantee the performance of AutoⁿML's "top" pipeline on the test data, we collected metrics from the AutoⁿML's "best" pipeline per dataset, which outputs the highest AUC score on the test data. Therefore, for our framework evaluation, we consider the following five pipelines to represent the selected AutoML systems for further comparison, the top pipelines of Auto-Sklearn, TPOT AutoML, H₂O AutoML, and AutoⁿML, and the best pipeline of AutoⁿML.

A. Performance Over Time Budget

To illustrate performance over different time budget constraints, we plot the performance metrics on the test data (AUC-based Rank and Accuracy-based Lift) for 1-minute-, 10-minute-, and 20-minute training time limits. Each data point on Figure (1) and Figure (2) represents the average performance of an AutoML system for the given training time limit across all datasets. In addition, these figures illustrate the performance of the other AutoML systems tested relative to the performance of AutoⁿML.

In Figure (1), the AutoⁿML "best" pipeline remains the top-scoring pipeline in terms of average AUC ranking, re-

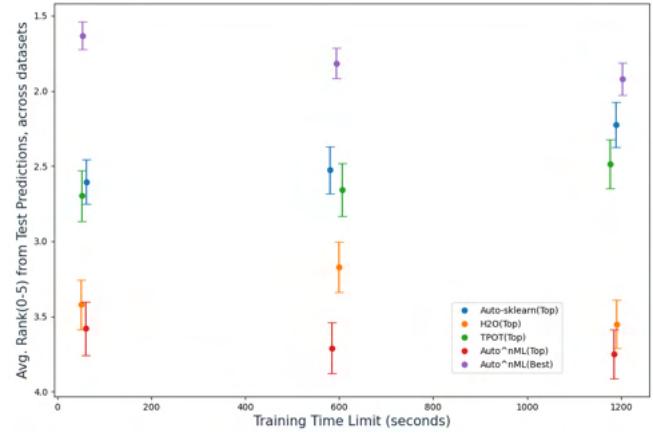


Fig. 1. Average Rank per AutoML on Test Predictions across datasets for each training time limit (1 minute, 10 minutes, 20 minutes)

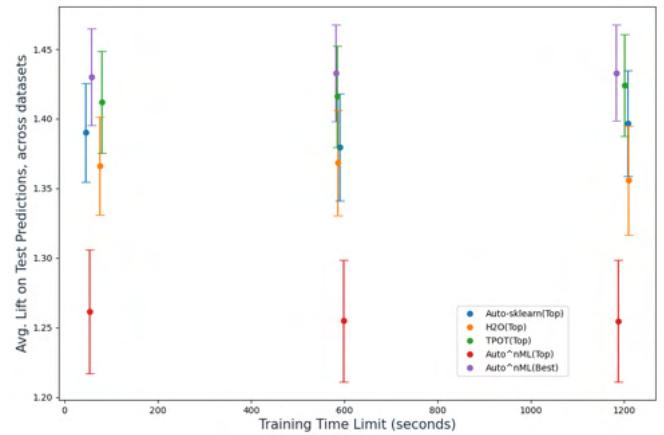


Fig. 2. Average Lift on Test Predictions across datasets for each training time limit (1 minute, 10 minutes, 20 minutes)

gardless of training time limit. For easier readability, the y-axis is inverted such that the highest ranking pipelines appear at the top. We observe that AutoⁿML "best" pipeline achieves the highest rank under the limited training time constraint of 1 minute, which is significantly better in comparison to those of other AutoML systems. Following the AutoⁿML "best" pipeline are Auto-Sklearn, TPOT, H₂O, and AutoⁿML "top" pipeline from highest-to-lowest rank. Unexpectedly, there exists a large discrepancy between the AutoⁿML "top" pipeline (ranked from training performance) and AutoⁿML "best" pipeline (ranked from test-data performance). The AutoⁿML "top" pipeline ranks lowest for all training time limits, and it ranks significantly lower than TPOT for the 10 minute training time limit. As the ranking is based on AUC, AutoⁿML's "top" pipeline has the lowest AUC score and the "best" pipeline has the highest AUC score on the test data on-average, over all training time limits. This is contrary to the fact that, during pipeline search, the "top" pipeline ranks above "best" pipeline by the same scoring metric of AUC.

As shown in Figure (2), the AutoⁿML "best" pipeline

TABLE I

LIFT AND RANK STATISTICS ON TRAINING DATA PER TRAINING TIMELIMIT, AVERAGED ACROSS N=76 DATASETS, \pm STANDARD ERROR OF THE MEAN

	1 minute		10 minutes		20 minutes	
	Avg. Rank	Avg. Lift	Avg. Rank	Avg. Lift	Avg. Rank	Avg. Lift
Auto-sklearn(Top)	1.95 \pm 0.10	1.52 \pm 0.04	1.91 \pm 0.09	1.49 \pm 0.04	1.91 \pm 0.09	1.50 \pm 0.04
H2O(Top)	2.37 \pm 0.12	1.50 \pm 0.03	2.32 \pm 0.12	1.47 \pm 0.04	2.39 \pm 0.11	1.47 \pm 0.04
TPOT(Top)	1.63 \pm 0.10	1.54 \pm 0.04	1.61 \pm 0.11	1.54 \pm 0.04	1.50 \pm 0.09	1.54 \pm 0.04
Auto^nML(Top)	3.90 \pm 0.07	0.89 \pm 0.06	3.86 \pm 0.07	0.89 \pm 0.06	3.92 \pm 0.07	0.89 \pm 0.06
Auto^nML(Best)	4.59 \pm 0.09	0.76 \pm 0.07	4.63 \pm 0.09	0.76 \pm 0.07	4.66 \pm 0.07	0.76 \pm 0.07

TABLE II

LIFT AND RANK STATISTICS ON TEST DATA PER TRAINING TIMELIMIT, AVERAGED ACROSS N=76 DATASETS, \pm STANDARD ERROR OF THE MEAN

	1 minute		10 minutes		20 minutes	
	Avg. Rank	Avg. Lift	Avg. Rank	Avg. Lift	Avg. Rank	Avg. Lift
Auto-sklearn(Top)	2.61 \pm 0.15	1.39 \pm 0.04	2.53 \pm 0.16	1.38 \pm 0.04	2.22 \pm 0.15	1.40 \pm 0.04
H2O(Top)	3.42 \pm 0.17	1.37 \pm 0.04	3.17 \pm 0.17	1.37 \pm 0.04	3.55 \pm 0.16	1.36 \pm 0.04
TPOT(Top)	2.70 \pm 0.17	1.41 \pm 0.04	2.66 \pm 0.18	1.42 \pm 0.04	2.49 \pm 0.16	1.42 \pm 0.04
Auto^nML(Top)	3.58 \pm 0.18	1.26 \pm 0.05	3.71 \pm 0.17	1.26 \pm 0.04	3.75 \pm 0.16	1.26 \pm 0.04
Auto^nML(Best)	1.63 \pm 0.09	1.43 \pm 0.04	1.82 \pm 0.10	1.43 \pm 0.04	1.92 \pm 0.11	1.43 \pm 0.03

achieves the highest average lift of accuracy on the test data. Following the Auto^nML “best” pipeline are TPOT, Auto-sklearn, H2O, and Auto^nML“top” pipeline from highest-to-lowest lift. Unlike Figure (1), the top four pipelines do not differ significantly in performance; Auto^nML “best” pipeline does not achieve significantly higher lift for any of the training time limits. Similar to Figure (1), though, the “top” pipeline shows significantly poor performance and the lowest average lift regardless of training time limit. As the lift is based on accuracy, this means that the “top” pipeline has the lowest accuracy and the “best” pipeline has the highest accuracy on the test data on-average, over all training time limits.

B. Training-Test Performance Discrepancies

To illustrate performance and relative training performance compared to test performance, we compare the performance metrics (accuracy-based lift and AUC-based rank) from the test data versus those from the training data. Each data point in Figure (3) and (4) represents performance of an AutoML system on a single dataset, averaged across the three time limits tested. Points along the diagonal at $y=x$ indicate identical training- and test- data performance.

Figure (3) indicates discrepancies between the rank of our Auto^nML on training data and on testing data. The plot compares the rank metric on training data versus that on test data for each AutoML system. On this plot, the most ideal performance is toward the upper right, which indicates that an AutoML pipeline has the highest rank (based on AUC scores) on both the training and test datasets. We observe that Auto^nML’s “top” and “best” pipelines have a higher AUC score on the test data than on the training data for the majority of data tasks, which are those data points lie above the $y=x$ line. The other AutoML systems achieve higher AUC scores on the training data than on the unseen test data, which are data points that fall below the $y=x$ line. It is expected that models which fit to (and predict well on) the training data are subject to perform relatively poorly on unseen test data.

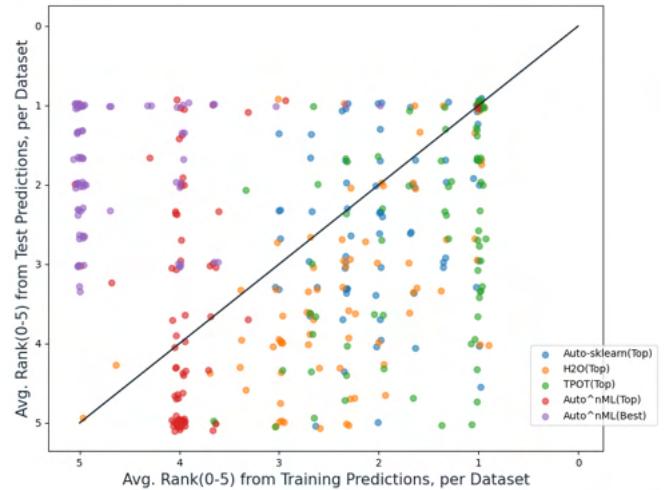


Fig. 3. Average Rank per AutoML (based on AUC score) on Test Predictions versus Average Rank of each AutoML (based on AUC score) on Training Predictions

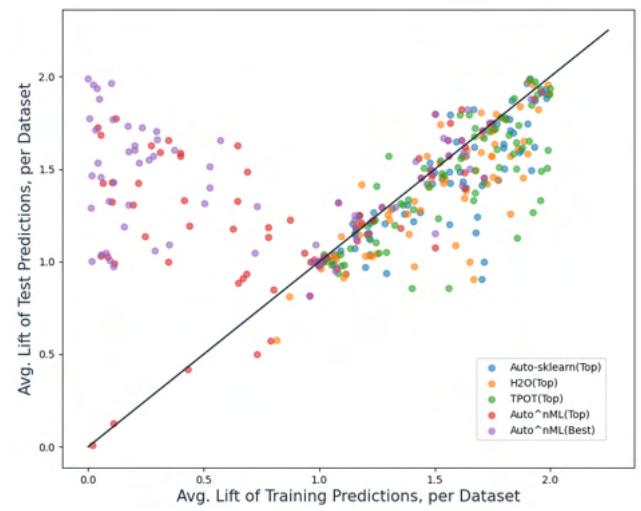


Fig. 4. Average Lift (based on accuracy) of Test Predictions versus Average Lift (based on accuracy) of Training Predictions per dataset

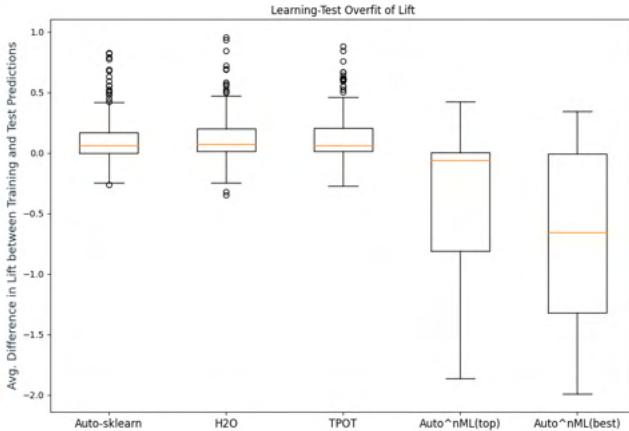


Fig. 5. Box and Whisker plot of the average difference between lift on training-data predictions and lift on test-data predictions

In addition, we observe a cluster of datasets in the upper-left, for which AutoⁿML’s “best” pipeline ranks first on the test dataset and ranks lowest on the training dataset. There also exists a cluster of datasets at the lower left, for which AutoⁿML’s “top” pipeline ranks lowest for both training and test datasets.

Figure (4) further illustrates the poor performance of AutoⁿML on the training dataset than on the test dataset. AutoⁿML “top” and “best” pipelines achieve higher lift (based on accuracy) on the test data than on the training data. For the majority of datasets evaluated, data points of the AutoⁿML’s “best” pipeline lie above the $y=x$ line. Similar to Figure (3), the data points of performance of Auto-Sklearn, TPOT, and H2O fall below the line $y=x$; this indicates greater accuracy on the training dataset than on the unseen test dataset. AutoML systems with the most ideal performance are toward the upper right, with high lift on the training and test datasets. Very few of the “best” pipeline’s data points lie in the upper right of the plot, indicating high accuracy on the training and test dataset predictions. Instead, we observe a cluster of datasets in the upper-right, for which other AutoML systems (such as TPOT) have high training and test accuracy.

To summarize the performance discrepancies, the box-and-whisker plots in Figure (5) show the median and quartiles of the lift statistics, as opposed to averaged lift statistics in Figure (4). The y-axis is the difference between the lift on training data and the lift on the test data, where ideal performance is a difference close to 0. For more than half of datasets provided to TPOT, Auto-Sklearn, and H2O, the “training lift” is greater than “test lift” per dataset. In contrast, for more than half of the datasets provided to either “top” or “best” pipeline of Auto-Sklearn, test lift is greater than training lift. In particular, the “best” pipeline of Auto-Sklearn is more likely to have greater accuracy on the test dataset, as the median of “lift” difference is below 0. Although the TPOT, Auto-Sklearn, and H2O boxplots show several outliers, these outliers are within a smaller range compared to the range of data points of the AutoⁿML

boxplots. Therefore, there is greater variance between the training and test lift of AutoⁿML pipelines where some datasets have severely poor training performance compared to test performance.

C. Dependency of Performance on Dataset

To illustrate the performance of each AutoML system based on size and type of data task, we explored the correlation between performance metrics (AUC-based Rank and Accuracy-based Lift) and various meta features of the input datasets. In Figure (6) and Figure (7), we plot the performance metrics versus the dimensionality (number of columns) of the input datasets. Each data point represents the performance of an AutoML system on a single dataset at the 20-minute training time limit. Figure (6) compares the AUC ranks of all AutoML systems on the test data, and Figure (7) compares the lift scores in accuracy on test data from “best” and “top” pipelines of AutoⁿML.

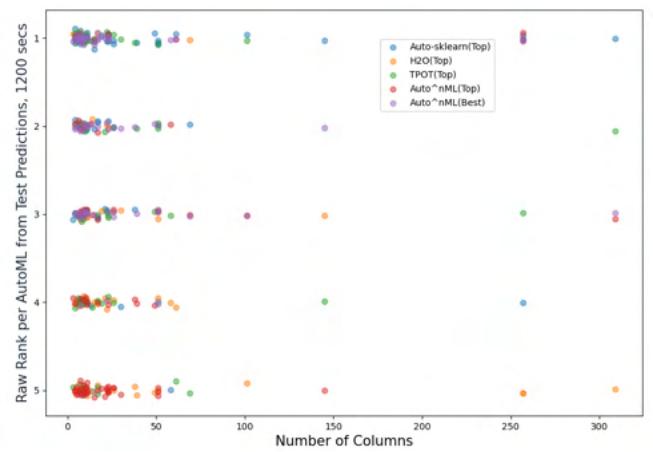


Fig. 6. Raw rank (based on AUC score) of each AutoML system at 20 minute training time limit versus the number of columns per input dataset

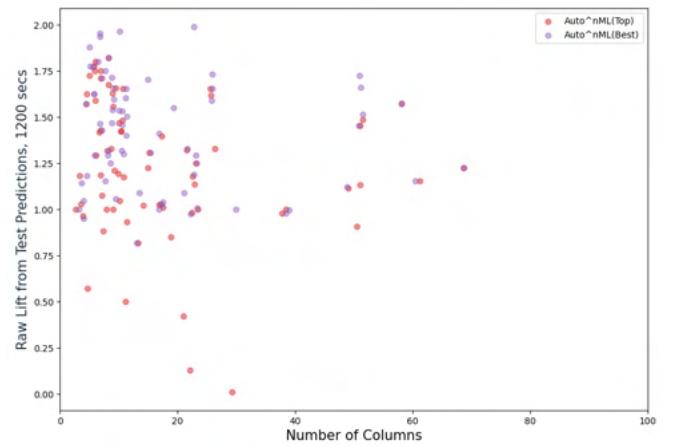


Fig. 7. Raw lift (based on accuracy) of “best” and “top” AutoⁿML pipelines at 20 minute training time limit versus the number of columns per the input dataset

By visual inspection of Figure (6), the AutoⁿML “best” pipelines achieve the highest ranks on the test datasets for

various dimensionalities. We observe a cluster of datasets in the upper left of the graph that AutoⁿML “best” pipeline outperforms other pipelines for low dimensional data with fewer than 50 features. By visual inspection of Figure (7), AutoⁿML “best” pipelines show unexpected poor lift on low dimensional datasets, some of which have lift below 1, indicating that those pipelines fail to beat the baseline/majority-vote classifier model accuracy.

Figure (6) and Figure (7) do not illustrate a clear linear relationship between performance and various features of the input space. To explore how AutoML performance metrics are related to the meta features of the datasets, we devised five binary classification tasks. We first compiled a list of meta features and outcomes from our framework evaluation over all 76 OpenML datasets under three training time limits (228 data points in total). For each classification task, we will train Random Forest models on meta features of the input dataset to predict whether an outcome related to the performance of AutoⁿML pipelines will occur (‘1’) or will not occur (‘0’). The target outcomes are defined as follows:

- **AutonTopWins-Rank:** the “top” pipeline outperforms other pipelines by AUC-based Rank on test data
- **AutonTopWins-Lift:** the “top” pipeline outperforms all other pipelines by accuracy-based Lift on test data
- **AutonBestWins-Lift:** the “best” pipeline outperforms other pipelines by accuracy-based Lift on test data
- **TopAndBestPipelinesIdentical:** the “top” AutoⁿML pipeline found during training is the same pipeline as the “best” AutoⁿML pipeline on the test dataset

From the best model per classification task, we also explored the feature importance and the distribution of the top features across different class labels to get insights on which datasets our AutoⁿML might outperform others and which datasets that our AutoⁿML has poor performance.

As shown in Table (III), we train three models with varying selections of input dataset features. The first model is trained using OpenML dataset metadata only— such as number of rows (‘NumberofInstances’), number of columns (‘NumberofFeatures’), and number of missing values (‘NumberofMissingValues’), and the accuracy of predicting the majority class on all datavalues (‘MajorityClassAccuracy’). The second model is trained using the metadata features and a binary feature derived from whether the top pipeline ranked from training data performs the best on test data (‘TopAndBestPipelinesIdentical’). The third model is trained using the metadata features and training time limits, in seconds (‘Timelimit(s)’). The best model for each classification task is selected by highest AUC score, marked in bold in Table(III).

The feature importances of these best models are shown in Table (IV). For every task, the top three important features for prediction derive from the meta features of the input dataset alone: including ‘MajorityClassAccuracy’, ‘NumberofInstances’, and ‘NumberofFeatures’. Additional statistics obtained after running our framework evaluation such as ‘TopAndBestPipelinesIdentical’ and ‘Timelimit(s)’ are comparatively unimportant. Next, we will analyse the

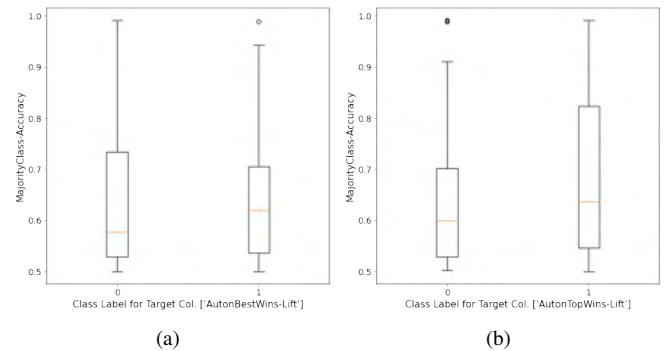


Fig. 8. Distribution of ‘MajorityClassAccuracy’ for (a) ‘AutonBestWins-Lift’ and (b) ‘AutonTopWins-Lift’ Classification Tasks

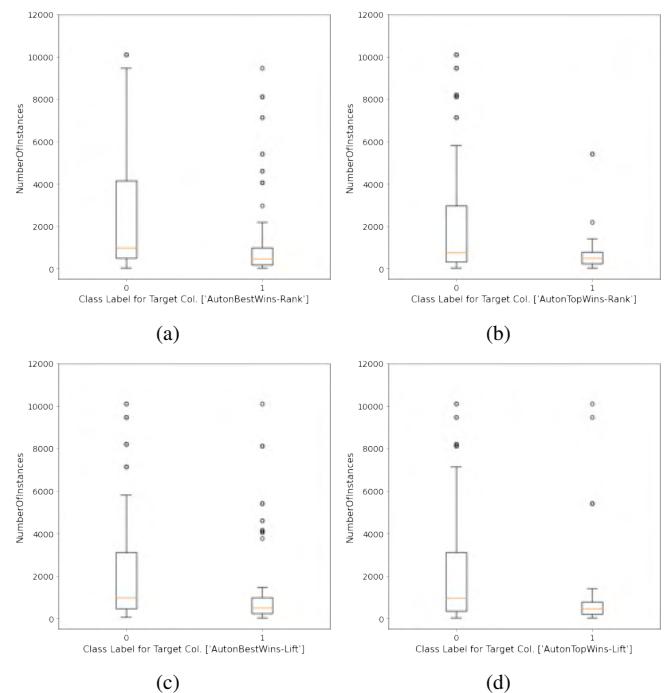


Fig. 9. Distribution of ‘NumberofInstances’ for (a) ‘AutonBestWins-Rank’, (b) ‘AutonTopWins-Rank’, (c) ‘AutonBestWins-Lift’, and (d) ‘AutonTopWins-Lift’ Classification Tasks

value distribution of the top three most important features within the ‘0’ and ‘1’ classes of each classification task:

1) ‘MajorityClassAccuracy’: In Figure (8), class=1 represents experiments for which “top” or “best” pipelines achieve the highest lift among all pipelines. In either plot, the distribution of ‘MajorityClassAccuracy’ between two classes is not significantly different.

2) ‘NumberofInstances’: Figures in (9) indicate that AutoⁿML may have higher chance to outperform other frameworks by both lift and rank on smaller datasets with fewer number of instances, but not outperform others on large datasets.

TABLE III
AUC SCORE FOR EACH MODEL (3) WITH VARYING SELECTIONS OF INPUT DATASET FEATURES, PER CLASSIFICATION TASK (5)

		Input: metadata only [228,4]	Input: metadata & identical pipelines condition [228,5]	Input: metadata & training timelimits [228,5]
Target Task	[TopAndBestPipelinesIdentical]	1.0000	-	0.9605
	[AutonTopWins-Rank]	0.9886	0.9837	0.9902
	[AutonTopWins-Lift]	0.8734	0.9213	0.8851
	[AutonBestWins-Rank]	0.8093	0.8043	0.8099
	[AutonBestWins-Lift]	0.8750	0.8894	0.8300

TABLE IV
TOP THREE FEATURES OF BEST-MODELS FOR EACH CLASSIFICATION TASK, RANKED HIGHEST TO LOWEST IMPORTANCE

[TopAndBestPipelinesIdentical]	MajorityClass-Accuracy (0.367), NumberOfFeatures (0.322), NumberOfInstances (0.262)
[AutonTopWins-Rank]	MajorityClass-Accuracy (0.316), NumberOfInstances (0.307), NumberOfFeatures (0.287)
[AutonTopWins-Lift]	MajorityClass-Accuracy (0.333), NumberOfInstances (0.316), NumberOfFeatures (0.219)
[AutonBestWins-Rank]	NumberOfInstances (0.326), NumberOfFeatures (0.247), MajorityClass-Accuracy (0.241)
[AutonBestWins-Lift]	MajorityClass-Accuracy (0.344), NumberOfInstances (0.340), NumberOfFeatures (0.218)

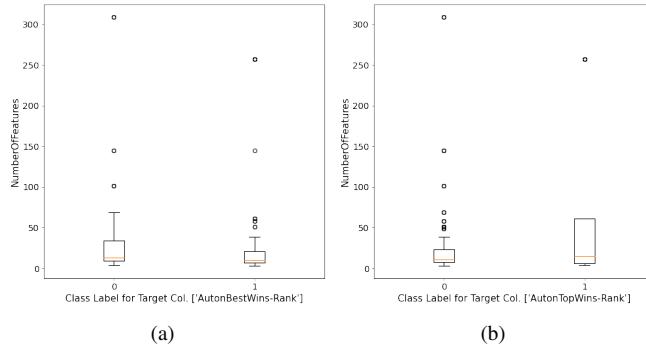


Fig. 10. Distribution of 'NumberOfFeatures' for (a) 'AutonBestWins-Rank' and (b) 'AutonTopWins-Rank' Classification Tasks

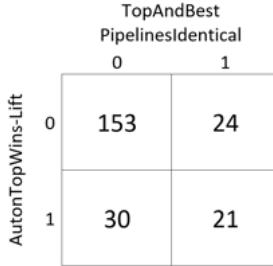


Fig. 11. Contingency Table between the number of experiments in which Auto^nML's top pipeline outperforms all pipelines, and the number of experiments in which both Auto^nML pipelines are identical

3) ‘NumberOfFeatures’: In Figure (10), the Auto^nML “best” pipeline outperforms the pipelines of other frameworks for relatively lower-dimensional data with 25 or fewer features. In contrast, the Auto^nML “top” pipeline is likely to outperform other pipelines over a wider range of dimensionalities– even for datasets with 50-70 features.

An interesting result is that the feature ‘TopAndBestPipelinesIdentical’ did not rank among the top 3 features for prediction of any of the five classification tasks. Although ‘TopAndBestPipelinesIdentical’ is not a meta feature of the input dataset, it is interesting to determine if information

of whether the “top” and “best” pipelines are identical (combined with information on the meta features of the dataset) can predict the relative performance of Auto^nML in a given experiment. Figure (11) illustrates a contingency table of the feature of ‘TopAndBestPipelinesIdentical’ versus outcomes of the classification task ‘AutonTopWins-Lift’. The ‘1’ class of the ‘AutonTopWins-Lift’ target column represents experiments for which the Auto^nML “top” pipeline outperforms other pipelines by Lift. In element [‘0’,‘0’], we observe that for the majority of experiments that the “top” pipeline fails to outperform others, the “top” and “best” pipelines are not identical. For experiments that the “top” pipeline outperforms others, row [‘1’], there is an equal likelihood that the “top” and “best” pipelines are identical or different.

V. DISCUSSION

In most evaluations, our Auto^nML framework outperforms TPOT, Auto-Sklearn and H2O AutoML, considering the performance of Auto^nML’s “best” pipeline, or the pipeline with the highest AUC score on the test data. Under the current version of Auto^nML, however, the end-user would typically apply the “top”-ranking pipeline with the highest AUC score on the training data for prediction on the unseen test data, which performs the worst compared to the other publicly-available AutoML systems evaluated. An in-depth review of pipeline ranking mechanism from training data is needed given our observations. Furthermore, we have found an interesting result that the “top” pipeline is more likely to perform poorly when the “top” and “best” pipelines differ.

In our analyses, we also identified features of input datasets for which our Auto^nML would likely outperform, or perform poorly compared to other frameworks. Results of our classification task suggest that Auto^nML pipelines have the best performance on smaller-size input datasets, of 1000 or fewer rows. Auto^nML pipelines also appear to have the best overall model performance (AUC) on lower-dimensional datasets, of 25 or fewer features.

A. Future Work

For a minority of datasets, AutoⁿML the “top” and “best” pipelines found during training and testing, respectively, are identical. In future work, we need to review the pipeline ranking process and ensure that the “best” pipeline could be generalized with robust performance on testing data. We also need to diagnose why the AUC scores during training fall below 0.1, which might cause the “top”-ranking pipeline to overshadow the true “best” pipeline for prediction. Furthermore, we also want to identify the source of discrepancies between training-and-test performance within our system.

By understanding features of the data tasks which influence low AUC scores during training, we hope to find ways to mitigate selecting the wrong pipeline. As an extension to this project, we may revise our classification tasks, which use meta features of a dataset to predict whether the AutoⁿML “top” pipeline will win amongst all AutoML systems, whether the AutoⁿML “best” pipeline will win amongst all AutoML systems, or whether the AutoⁿML “top” pipeline found during training will be identical to the “best” pipeline for prediction

The current experiments and findings are limited to only 76 OpenML classification datasets. Thus, the findings might not thoroughly represent other datasets and various data tasks such as regression. Future work could focus on experiments using a wider range of datasets and data tasks, as well as longer training time constraints.

APPENDIX

Please See Table (V), attached, for summary of average performance metrics and metadata features per OpenML dataset evaluated.

ACKNOWLEDGMENT

This work is supported by National Science Foundation (NSF) Grant #1659774 and Carnegie Mellon Robotics Institute Summer Scholars (RISS) program. Thank you to RISS program organizers (Rachel Burcin and Dr. John Dolan) and my mentors (Jieshi Chen and Dr. Artur Dubrawski) for their consistent encouragement and support, especially considering the virtual nature of this internship.

REFERENCES

- [1] M.-A. Zöller and M. F. Huber, “Benchmark and Survey of Automated Machine Learning Frameworks,” *arXiv:1904.12054 [cs, stat]*, Jan. 2021. [Online]. Available: <http://arxiv.org/abs/1904.12054>
- [2] J. Yoo, T. Joseph, D. Yung, S. A. Nasseri, and F. Wood, “Ensemble Squared: A Meta AutoML System,” *arXiv:2012.05390 [cs]*, June 2021, arXiv: 2012.05390. [Online]. Available: <http://arxiv.org/abs/2012.05390>
- [3] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li, and A. Smola, “AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data.”
- [4] E. LeDell and S. Poirier, “H2O AutoML: Scalable Automatic Machine Learning.”
- [5] J. Thomas, S. Coors, and B. Bischof, “Automatic Gradient Boosting,” *arXiv:1807.03873 [cs, stat]*, July 2018, arXiv: 1807.03873. [Online]. Available: <http://arxiv.org/abs/1807.03873>
- [6] M. Milutinovic, B. Schoenfeld, and D. Martinez-Garcia, “On Evaluation of AutoML Systems.”

TABLE V

PERFORMANCE METRICS OF EACH AUTOML SYSTEM, AVERAGED ACROSS TRAINING TIMELIMITS, AND METADATA PER OPENML DATASK

OpenML-ID	No. of Instances	No. of Features	No. of Missing Values	MajorityClass-Accuracy	AutonTop-AvgRank	AutonTop-AvgLift	AutonBest-AvgRank	AutonBest-AvgLift
15	699	10	16	0.6743	4.333	1.424	1.333	1.458
24	8124	23	2480	0.5022	5.000	1.000	1.000	1.991
27	368	23	1927	0.6630	4.667	1.180	1.667	1.295
31	1000	21	0	0.7040	5.000	0.420	1.000	1.091
37	768	9	0	0.6406	4.333	1.211	2.000	1.252
44	4601	58	0	0.6090	2.333	1.572	1.333	1.576
50	958	10	0	0.6458	5.000	1.194	3.333	1.535
56	435	17	392	0.6697	2.333	1.397	1.000	1.411
151	45312	9	0	0.5775	4.667	1.333	1.667	1.597
334	601	7	0	0.6821	5.000	0.883	2.667	1.466
466	340	15	834	0.5176	5.000	1.227	2.667	1.705
715	1000	26	0	0.5240	5.000	1.328	1.667	1.733
717	508	11	0	0.5748	5.000	1.425	3.000	1.534
723	1000	26	0	0.5680	4.333	1.620	2.667	1.592
728	4052	8	0	0.7720	5.000	1.000	1.000	1.293
729	44	4	0	0.6364	1.000	1.571	1.000	1.571
732	250	51	0	0.5238	4.333	1.485	1.333	1.515
733	209	7	0	0.6604	2.000	1.429	1.000	1.429
737	3107	7	0	0.5032	3.000	1.711	3.000	1.711
740	1000	11	0	0.6200	4.000	1.484	2.667	1.503
750	500	8	0	0.5280	1.000	1.318	1.000	1.318
755	62	6	0	0.5000	3.333	1.750	1.000	1.667
757	528	22	504	0.8864	5.000	0.128	2.000	0.974
758	67	15	0	0.7647	1.000	1.308	1.000	1.308
761	8192	22	0	0.7021	5.000	1.323	3.333	1.330
766	500	51	0	0.5440	5.000	1.132	3.000	1.662
767	475	4	0	0.8908	4.333	1.028	2.000	1.047
770	625	7	0	0.5159	5.000	1.185	1.000	1.938
792	500	6	0	0.5200	1.333	1.800	1.333	1.800
798	303	14	6	0.5789	2.333	1.023	1.000	1.091
799	1000	6	0	0.5360	5.000	1.590	3.000	1.627
803	7129	6	0	0.5311	5.000	1.776	1.333	1.772
805	500	51	0	0.6000	3.000	1.453	3.000	1.453
823	20640	9	0	0.5723	4.667	1.685	3.000	1.714
839	782	9	466	0.6122	2.333	1.558	1.000	1.542
841	950	10	0	0.5042	5.000	1.658	2.000	1.967
842	60	11	14	0.6667	4.667	0.500	1.667	1.300
873	250	51	0	0.5238	5.000	0.909	2.333	1.727
878	100	11	0	0.6000	4.667	0.933	1.000	1.400
903	1000	26	0	0.5560	2.000	1.655	2.000	1.655
913	1000	11	0	0.5680	2.000	1.655	2.000	1.655
936	500	11	0	0.5440	5.000	1.176	2.000	1.603
942	50	4	0	0.5385	5.000	0.571	2.000	1.048
945	76	7	0	0.6842	2.667	1.077	1.667	1.154
966	1340	17	20	0.9104	5.000	1.030	1.000	1.039
967	406	9	14	0.6275	3.000	1.469	3.000	1.469
968	365	4	30	0.9022	5.000	0.964	1.333	0.952
981	10108	69	2699	0.7341	3.000	1.226	3.000	1.226
983	1473	10	0	0.5637	5.000	1.048	2.000	1.317
987	500	23	0	0.8000	1.000	1.250	1.000	1.243
997	625	5	0	0.5287	5.000	1.627	2.333	1.880
1004	600	61	0	0.8667	1.000	1.154	1.000	1.154
1006	148	19	0	0.5405	5.000	0.850	1.000	1.550
1011	336	8	0	0.5357	3.000	1.822	1.000	1.822
1013	138	3	0	0.9429	3.667	1.000	1.000	1.000
1055	89	9	0	0.7391	5.000	1.000	1.000	1.059
1056	9466	39	0	0.9920	4.000	1.000	2.000	0.999
1073	274	9	0	0.5072	3.667	1.629	1.000	1.657
1444	1043	38	0	0.8851	4.000	0.978	2.333	0.978
1461	45211	17	0	0.8845	3.667	1.026	2.333	1.027
1462	1372	5	0	0.5627	5.000	1.725	1.000	1.775
1479	1212	101	0	0.5149	3.000	1.455	3.000	1.455
1488	195	23	0	0.7551	5.000	1.135	1.000	1.189
1489	5404	6	0	0.7054	1.333	1.295	1.333	1.295
1490	182	13	0	0.7174	2.000	0.818	2.000	0.818
1495	250	7	0	0.5714	1.000	1.750	1.000	1.750
1506	470	17	0	0.8475	2.000	1.010	1.000	1.000
40704	2201	4	0	0.6388	1.667	1.185	1.667	1.185
40713	3772	30	0	0.9894	5.000	0.011	2.333	1.001
40922	88588	7	0	0.5059	4.333	1.588	2.333	1.956
41143	2984	145	0	0.5054	5.000	1.000	1.667	1.605
41145	5832	309	0	0.5117	3.000	1.497	3.000	1.497
41156	4147	49	0	0.7753	3.667	1.114	2.000	1.123
41160	31406	23	29756	0.9019	5.000	0.989	2.667	1.008
41964	1424	257	0	0.5140	1.000	1.918	1.000	1.918
41973	319	257	0	0.5250	1.000	1.881	1.000	1.881

OpenML-ID	Autosk-AvgRank	Autosk-AvgLift	H2O-AvgRank	H2O-AvgLift	TPOT-AvgRank	TPOT-AvgLift
15	2.667	1.421	3.667	1.404	2.333	1.407
24	1.000	1.991	1.000	1.978	1.000	1.987
27	1.667	1.219	4.333	1.148	2.667	1.000
31	3.000	1.038	4.000	1.044	2.000	1.047
37	3.000	1.149	4.667	1.198	1.000	1.228
44	4.000	1.571	3.000	1.569	4.333	1.560
50	2.000	1.484	2.000	1.533	1.000	1.548
56	2.667	1.402	4.000	1.384	5.000	1.374
151	3.667	1.486	1.333	1.590	3.667	1.478
334	1.667	1.450	4.000	1.424	1.000	1.466
466	2.333	1.439	4.000	1.447	1.000	1.848
715	2.333	1.435	2.667	1.735	3.333	1.718
717	3.333	1.484	2.000	1.493	1.667	1.511
723	2.333	1.615	4.667	1.573	1.000	1.643
728	3.000	1.286	4.000	1.283	2.000	1.192
729	1.000	1.571	1.000	1.571	1.000	1.571
732	3.333	1.414	4.333	1.343	1.667	1.495
733	4.000	1.267	2.667	1.419	4.667	1.410
737	1.000	1.744	4.333	1.678	2.667	1.736
740	1.667	1.503	5.000	1.458	1.667	1.503
750	4.333	1.157	4.333	0.934	3.333	1.268
755	2.000	1.708	4.000	1.000	4.333	1.333
757	2.333	0.989	4.000	0.997	1.667	0.983
758	1.000	1.308	1.000	1.256	1.000	1.308
761	1.000	1.121	2.000	1.327	3.667	1.335
766	1.333	1.696	3.333	1.583	2.333	1.652
767	3.000	1.060	4.667	1.031	1.000	1.072
770	2.000	1.885	2.000	1.909	1.000	1.938
792	2.667	1.795	4.667	1.744	4.000	1.795
798	4.667	0.970	3.000	1.030	4.000	1.053
799	2.000	1.622	2.000	1.654	3.000	1.617
803	2.667	1.769	4.000	1.778	2.000	1.780
805	1.000	1.493	5.000	1.258	2.000	1.489
823	1.333	1.717	1.667	1.715	4.333	1.705
839	4.000	1.533	2.667	1.492	5.000	1.506
841	2.667	1.964	3.000	1.956	2.333	1.972
842	3.333	1.200	3.333	1.033	1.667	1.233
873	2.333	1.646	4.000	1.444	1.333	1.677
878	3.667	1.244	2.667	1.089	3.000	1.333
903	3.000	1.420	5.000	1.573	2.000	1.609
913	5.000	1.620	3.333	1.599	1.667	1.667
936	1.000	1.637	3.667	1.422	3.333	1.583
942	2.333	0.905	3.333	0.905	2.333	0.857
945	5.000	1.000	3.667	0.974	2.000	1.128
966	3.333	1.039	3.000	1.031	2.667	1.040
967	3.333	1.427	3.333	1.464	1.333	1.510
968	3.000	0.936	3.667	0.916	2.000	0.940
981	1.667	1.150	1.333	1.226	5.000	1.218
983	3.333	1.152	3.000	1.277	1.667	1.316
987	1.000	1.250	3.667	1.207	1.000	1.250
997	1.333	1.871	4.000	1.799	1.667	1.884
1004	1.000	1.154	4.333	1.144	4.667	1.131
1006	1.667	1.267	3.333	1.450	3.667	1.683
1011	3.333	1.822	2.667	1.807	5.000	1.793
1013	1.667	1.000	2.667	1.000	5.000	1.000
1055	3.000	1.000	3.667	1.020	2.333	1.039
1056	2.333	1.001	4.000	0.997	2.667	1.000
1073	2.333	1.210	3.333	1.571	4.667	1.429
1444	3.000	1.000	4.333	0.812	1.333	0.978
1461	3.000	1.008	1.000	1.005	5.000	1.020
1462	1.000	1.777	3.000	1.769	1.000	1.772
1479	1.000	1.932	5.000	1.840	1.333	1.917
1488	2.000	1.207	4.000	1.144	2.667	1.198
1489	3.667	1.257	5.000	1.261	2.667	1.282
1490	4.000	0.939	5.000	0.576	1.000	0.859
1495	1.000	1.750	1.000	1.704	1.000	1.722
1506	3.333	0.980	4.333	0.963	4.333	0.983
40704	2.000	1.205	3.667	1.138	4.333	1.175
40713	3.333	0.997	3.333	1.002	1.000	1.000
40922	3.333	1.744	1.000	1.958	4.000	1.931
41143	1.333	1.606	3.000	1.614	4.000	1.602
41145	1.000	1.593	4.000	1.454	3.000	1.508
41156	1.000	1.072	4.667	1.030	3.667	1.105
41160	2.000	1.002	1.667	1.014	3.667	1.010
41964	3.667	1.905	5.000	1.823	3.333	1.903
41973	1.000	1.873	5.000	1.635	1.000	1.905

Ensembles for Online Dynamics Modeling of Off-road Terrain

Matthew Sivaprakasam¹, Samuel Triest², Wenshan Wang², Sebastian Scherer²

Abstract—Planning for autonomous ground vehicles has improved greatly over the past decades. They have become effective by using algorithms that take into account inherent structure and geometry in the surrounding environment. However, the same techniques are more likely to fail when brought off-road where there is less structure and complex terrain. There is still a need for planning techniques that effectively handle physical interactions between a vehicle and its surroundings. Motion planning for off-road terrain often employs some sort of dynamics model specific to a given robot, but these often only include properties specific to the robot itself, and don’t account for the fact that these properties are affected by the robot’s physical environment. We propose an approach which includes an ensemble of classical models. These dynamics models are generated offline in a semi-realistic simulation that accounts for a wide array of physical environmental properties. A higher-level model is then trained online to choose which dynamics model to use given the performance of previous decisions. This approach results in more accurate state predictions compared to the baseline in simulation in complex terrain.

Index Terms—Learning from Experience, Dynamics, Model Learning for Control

I. INTRODUCTION

Path planning and control for robots has made great strides in the past years. In planning, hand-tuned costs can be designed to take advantage of structure and geometry in an environment in order to find an optimal path [1]. In order to ensure that a path is kinodynamically feasible, some planning algorithms will also employ some sort of vehicle model that determines what states a robot can reach from its current state. A controller can then use the same model when outputting commands in order to track the desired path. This general approach tends to work well for organized and structured environments [2], [3]. For example, autonomous driving in urban environments is a relatively constrained problem, when it comes to physical interactions with the environment. Due to urban infrastructure, there is a limited number of types of surfaces a car is expected to drive on. Each of these surfaces are generally consistent in terms of physical properties like friction and deformability, which means that simpler models can be used to predict how a vehicle will respond on them.

The consistencies taken for granted in structured environments are often unknown and erratic in off-road environments. Due to obstacles like unpaved surfaces, shrubbery, rocks, and deformable objects, the physical properties of the

ground can change even within the span of a meter. This makes it much more difficult to use a single simple model to make predictions the same way it might be used in a simpler environment, as the optimal model changes as the environment changes. Moreover, it is difficult to determine the best model prior to deployment without having extensive knowledge of the given environment beforehand.

In order to use kinodynamic planners and controllers effectively in off-road environments, accurate vehicle modeling is needed. Complex models can be used, for example neural networks, but they require extensive amounts of diverse data in order to be robust to a variety of terrains. Additionally, they rely primarily on data collected prior to deployment rather than on data collected in real-time. In this work we instead propose an ensemble comprised of simple models, that learns online in order to make effective predictions. Transfer functions to predict velocity and steering angle were trained on different types of surfaces, and each model uses one of them to predict the next state of the robot. The effectiveness of each expert in previous decisions is taken into account when making the next decision. By updating the model online, we demonstrate higher accuracy in state predictions compared to the baseline of using a single static model.

II. RELATED WORK

There are several prior works related to the deployment of robots in off-road environments. Xiao *et al.* introduced a data-driven kinodynamic planner for fast off-road navigation [4]. They learn a neural net that takes in IMU information and a desired state and outputs control commands. It takes in no vision-based perception (e.g. lidar, camera), and instead train the inverse kinodynamics model on the IMU data alone. Their work establishes the fact that physical changes in an environment are hard to model by hand but can still significantly affect the dynamics of a vehicle. While Xiao’s work employs a model trained offline, online approaches are also becoming more common [5]–[7]. Work by Kumar *et al.* introduces a framework with both an offline and an online phase [8]. The first phase involves learning an encoder to map environment properties (for example friction and terrain height), and then using that as an input for model-free reinforcement learning. The goal of the online component is to map the previous states and actions to the latent vector since physical properties of the environment aren’t easily available outside of simulation. By using the different inputs for the online component they are able to address some of the problems that arise with sim-to-real transfer. There are also a number of papers that use learned dynamics models

¹Matthew Sivaprakasam is with the University of Pittsburgh, Pittsburgh, PA 15213, USA. mjs299@pitt.edu

²Samuel Triest, Wenshan Wang, and Sebastian Scherer are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA. {striest,wenshanw,basti}@andrew.cmu.edu

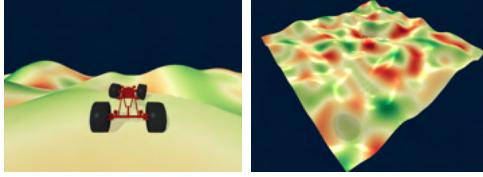


Fig. 1. The car used in simulation, and an example of a randomized terrain. Red areas correspond to lower friction and green areas correspond to higher friction.

directly [9], [10]. Work by Tremblay *et al.* proposes a way of supplementing prior state information with IMU, camera, and lidar information in order to improve dynamics predictions [11]. They treat each of the sensors as different experts, and learn how to prioritize each modality based on the state, which provides a promising method of indirectly extracting environmental information off-road. Xiao’s, Kumar’s, and Tremblay’s works all show the promise of using previous state information in making new decisions in unstructured environments.

III. METHODS

A. Physically-realistic Simulation Environment

Due to the necessity of realistic physics, a custom simulation environment was set up in pybullet [12]. The vehicle in simulation was a highly-configurable 4-wheeled robot with parameters such as max throttle, max steering angle, mass, friction, and suspension limits, damping, spring force. A configurable sensor suite was also implemented for the robot, including front-facing camera, IMU, lidar, shock travel (from suspension), heightmaps, and friction maps. Together, the accurate physics alongside thorough sensors are an attempt to reduce future sim-to-real transfer problems.

B. System Identification and Dynamics Modeling

In order to perform system identification, data was systematically collected in various simulation environments that were created by iterating through a range of slopes and frictions. In each environment, step responses were collected in a range of magnitudes for both commanded velocity and steering angle. This allowed us to model transfer functions that map from commanded to actual velocity and steering, which was accomplished by fitting third-order ARX (autoregressive with exogenous input) time-series models. These models, trained on the step response data using the L-BFGS optimization algorithm [13], maintain buffers of the past 2

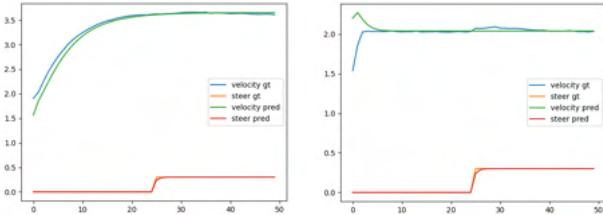


Fig. 2. Velocity and steering transfer functions collected on two surfaces with different friction.

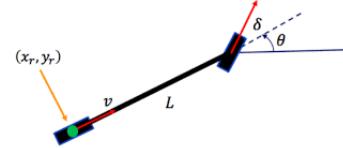


Fig. 3. Kinematic Bicycle Model

states and commands and use them along with the current state and command to output a predicted state. The transfer functions were generated systematically, so for N different friction values and M slope values in data collection, there are a resultant NxM total transfer functions each for velocity and steering angle as shown in Fig. 2.

For modeling the system dynamics, the kinematic bicycle model is used Fig. 3. The model has a three-dimensional state space comprising of location and heading in 2D space (using the rear axle as reference point). It has a single parameter L (vehicle length) and, given an input velocity and steering angle (yaw), predicts the change in state using the following update rules:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v * \cos(\theta) \\ v * \sin(\theta) \\ v * \tan(\delta)/L \end{bmatrix}$$

where x and y are horizontal and vertical position, θ is heading, v is velocity, and δ is yaw. To predict the next state of the robot, the input velocity and yaw are first predicted by feeding in the commanded and current velocity and yaw into their corresponding transfer functions. The predictions along with the current state are then fed into the bicycle model which outputs the next state.

As previously mentioned, all the transfer functions were generated in environments with different physical characteristics (slope and friction). Given the same input states, there is a variation in predictions across all the transfer functions, which impacts the predictions that come out of the bicycle model (Fig. 4). In theory, the variations together encompass the different possibilities of resultant states given any physical properties of the environment, provided that they are within the range of properties the ensemble of transfer functions themselves were trained on. Moreover, the set of transfer functions that results in the most accurate

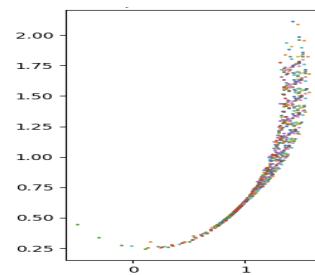


Fig. 4. Predictions using various transfer functions forward-sampled 10 steps into the future

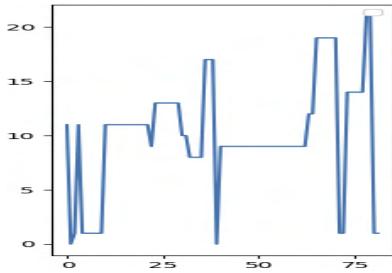


Fig. 5. The index of the best expert over time

predictions changes over time as a vehicle moves through complex terrain (Fig. 5).

C. Online Learning

In a "constant" environment, in which the physical properties are the same across the whole terrain, there would exist a single transfer function from the ensemble that always results in the best predictions. In this case, it would only take monitoring the predictions from the entire ensemble for a few steps to evaluate which transfer function has the highest performance. However, this strategy doesn't work in more realistic environments, where there are lots of variations in the terrain. Changes in the physical properties of the terrain means that the expert from the ensemble with the best performance also changes as the vehicle navigates the terrain (see Fig. 6). In order to address this problem, our method acts online by taking immediate feedback into consideration when choosing which transfer function to use to predict the next state.

Prior to the online component, an ensemble of experts is initialized offline. Each expert consists of a pair of transfer functions, one for velocity and one for yaw generated from a specific friction and slope, and a kinematics model (in this case they are all the same bicycle model). The experts take in an observed state (x, y, θ) and command (v, δ), feed it through their corresponding transfer function and models, and output a prediction for the next state. A vector of weights, one for each expert, is also initialized so that all experts start with the same weight.

The online component runs with each step in simulation. In a given step, all experts output a prediction based on the current state and command, and a pre-determined policy is

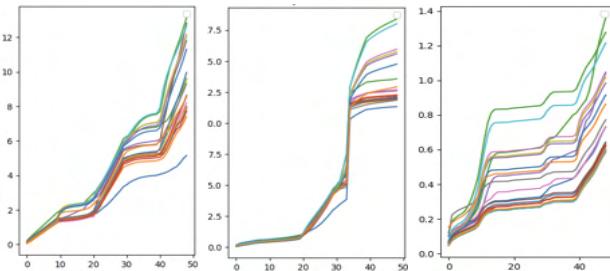


Fig. 6. The cumulative losses of various experts over the past 50 steps at different points in time. Loss is defined as the distance between the predicted state and the ground truth state.

used to select an expert based on their current weights. The next state is then observed, and a loss vector is generated by calculating the L2 distance between each expert's predicted state and the ground truth state (x, y, θ). This loss vector is used to update the weights vector based on the policy, and the new weights are used to make the decision in the next simulation step. For the Randomized Weighted Majority (RWM) policy [14], *all* weights are updated as follows:

$$W[i] = W[i] * e^{-\eta * l[i]} \quad (1)$$

where W is the weights vector, i is the index of the expert in the ensemble, η is a fixed hyper-parameter, and l is the loss vector. The weights are normalized to sum to 1 after each update. For the Generalized Weighted Majority (GWM) policy [14] only the weight for the chosen expert is updated:

$$W[i] = W[i] * e^{-\sqrt{\log(S/T)} * l[i]} \quad (2)$$

where S is the number of experts, and T is the total number of steps that have occurred. For the Exponential-weight algorithm for Exploration and Exploitation (EXP3) policy [15] again only the weight for the chosen expert is updated:

$$p = \frac{W[i]}{\sum W} \quad (3)$$

$$\eta = \sqrt{\frac{\log(S)}{T * S}} \quad (4)$$

$$W[i] = W[i] * e^{-\eta * l[i] / p} \quad (5)$$

All the policies implemented so far have the same decision process for choosing an expert, which involves using the weights as a probability vector and making a random decision based on those probabilities. However we did also use another strategy where, instead of choosing an expert, a new prediction is generated by taking the weighted average of all the expert predictions. These policies allow the learner to use the effectiveness of past predictions in the decision process for current predictions.

IV. RESULTS

As established earlier, there is not one transfer function that consistently results in the best predictions. Since the loss of each expert relative to the others changes over time (Fig. 6), the improvement our method provides can be indicated both by observing the distribution of expert predictions, as well as by comparing it to the baseline method (using a single transfer function for all predictions). Fig. 7 shows examples of the predictions of all experts in 2D space over time. As the vehicle moves across terrain in simulation, we observed that there are some areas with a small spread of predictions as well as areas with a wide distribution of predictions. Even when the distribution gets wider, the learner's predictions stay close to the ground truth, which shows the benefit of learning off of the varying loss of the experts relative to one another. For example, Fig. 8 shows how using a single

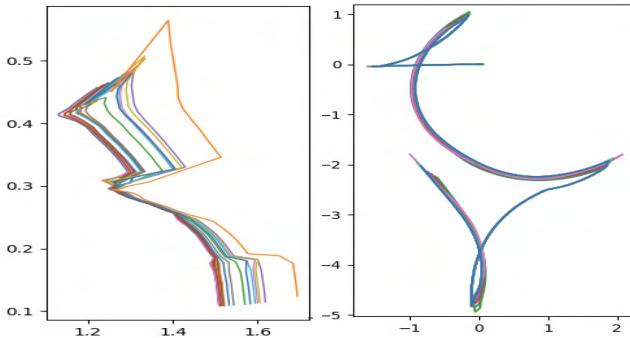


Fig. 7. Predictions of all experts in 2D space as the vehicle traverses terrain in two separate experiments.

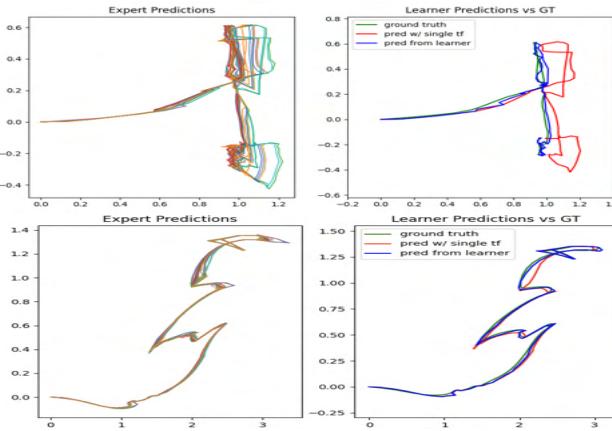


Fig. 8. Left: the distribution of all expert predictions in 2D space; Right: the accuracy of predictions from the learner compared to using a single transfer function.

transfer function for each prediction results in some areas with inaccurate predictions compared to using a learned prediction.

Additionally, the regret of the learner over time is observed, where regret is defined by the difference between the learner's cumulative loss and the minimum possible cumulative loss. Fig. 9 shows the regret of the learner over time in two different experiments. Decreasing regret values indicate good performance of the learner, since this corresponds to being as accurate as the best expert at any given time. The spikes of increased regret at various points correspond to changes in the terrain. This can be explained by the temporal nature of the learner, in that it will have higher weights for the expert that has been the most accurate recently. Therefore, when the terrain changes suddenly, the best expert changes drastically and the learner needs time to adjust accordingly.

Fig. 10 shows the performance of the different policies we experimented with. The more jagged loss of the GWM and EXP3 policies are likely explained by the fact that only one expert's weight is adjusted in each time step, compared to the smoother changes in loss from the RWM and weighted average resulting from all weights being updated at every step. Overall, the weighted average policy consistently outperformed the other policies.

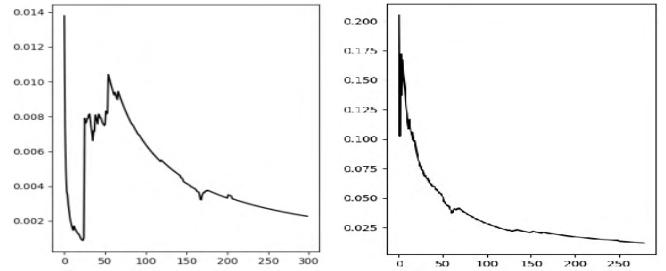


Fig. 9. The regret of the learner over time in two different experiments.

V. DISCUSSION AND FUTURE WORK

In this work we first establish that a wide range of transfer functions can be used to collectively represent the results of a vehicle driving in uneven terrain with variable friction, and that their individual performances vary as the terrain varies. We then demonstrate that online ensemble methods can be used to provide an improvement over using a single model by taking into account previous rewards when making new decisions. This behavior allows it to perform at least almost as well as the best expert, which varies greatly over time, at any given step.

Our immediate next step involves validating our method on a physical platform. We have an all-terrain vehicle (ATV), shown in Fig. 11 fitted with the same sensors that we setup our simulator. We plan to collect an ensemble of transfer functions on the ATV, and see if we observe the same improvement from online ensembles in order to show that our contribution has real-life applications.

Our method is also lightweight enough to integrate into downstream tasks, like planning and control. There are both planners, like kinodynamic RRT*, and controllers, like MPC, that work best when they have highly accurate vehicle models. They involve forward-sampling the vehicles state over multiple steps, so any error in the model ends up causing a drift in the predictions from the ground truth. In theory, the improvement that the online ensemble provides should help reduce this drift.

One limitation however, is its reliance on systematically collected data, which is difficult to accomplish in real-life. One way to address this is by collecting fewer transfer functions and just interpolating their parameters to artificially

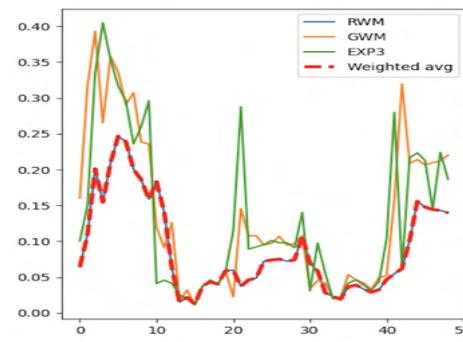


Fig. 10. The loss of the various learner policies over time.



Fig. 11. Our physical platform for real-life testing

generate more. However, it would probably be better to incorporate some approach that doesn't require any prior knowledge of the environment at all. This way, the ensemble would be agnostic to the range of properties in a given environment.

ACKNOWLEDGMENT

This work was supported by the Robotics Institute Summer Scholars program at Carnegie Mellon University. The authors would like to thank Rachel Burcin and Dr. John Dolan for their support and organization of this program.

REFERENCES

- [1] D. V. Lu, D. Hershberger, and W. D. Smart, "Layered costmaps for context-sensitive navigation," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 709–715.
- [2] D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Practical search techniques in path planning for autonomous driving," in *Proceedings of the First International Symposium on Search Techniques in Artificial Intelligence and Robotics (STAIR-08)*, 2008.
- [3] D. Ferguson and M. Likhachev, "Efficiently using cost maps for planning complex maneuvers," in *Proc. of the Workshop on Planning with Cost Maps, IEEE Int. Conf. on Robotics and Automation*, 2008.
- [4] X. Xiao, J. Biswas, and P. Stone, "Learning inverse kinodynamics for accurate high-speed off-road navigation on unstructured terrain," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 6054–6060, 2021.
- [5] M. Zambelli and Y. Demiris, "Online multimodal ensemble learning using self-learned sensorimotor representations," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 2, pp. 113–126, 2017.
- [6] R. Zou, Y. Liu, G. CHU, J. Zhao, and H. CAI, "Online active ensemble learning for robot collision detection in dynamic environments," *Journal of Mechanics in Medicine and Biology*, vol. 21, p. 2150035, 05 2021.
- [7] A. S. Panda, R. Prakash, L. Behera, and A. Dutta, "Combined online and offline inverse dynamics learning for a robot manipulator," in *2020 International Joint Conference on Neural Networks (IJCNN)*, 2020, pp. 1–7.
- [8] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," 2021.
- [9] N. R. Ke, A. Singh, A. Touati, A. Goyal, Y. Bengio, D. Parikh, and D. Batra, "Learning dynamics model in reinforcement learning by incorporating the long term future," 2019.
- [10] N. Takeishi and Y. Kawahara, "Learning dynamics models with stable invariant sets," 2020.
- [11] J.-F. Tremblay, T. Manderson, A. Noca, G. Dudek, and D. Meger, "Multimodal dynamics modeling for off-road autonomous vehicles," 2021.
- [12] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," <http://pybullet.org>, 2016–2021.
- [13] *Practical Methods of Optimization*. John Wiley & Sons, Ltd, 2000, pp. i–xvii. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118723203.fmatter>
- [14] N. Littlestone and M. Warmuth, "The weighted majority algorithm," *Information and Computation*, vol. 108, no. 2, pp. 212–261, 1994. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0890540184710091>
- [15] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2002. [Online]. Available: <https://doi.org/10.1137/S0097539701398375>

Multi-robot Simulation Dataset to Analyze Distributed Inference Under Perceptual Aliasing

Thomas Snyder¹, Sudharshan Suresh² and Michael Kaess³

Abstract—Distributed simultaneous localization and mapping (SLAM) allows multiple robots to map a larger area and develop a common understanding of the environment with a decreased single-robot workload. A persistent problem in SLAM algorithms is perceptual ambiguity; different scenes may induce similar sensory signals, causing the SLAM frontend to create erroneous constraints between nodes in the underlying pose graph. Identifying scenarios in which perceptual ambiguity occurs is important in evaluating algorithm resilience. In this paper, we simulate environments that may induce ambiguous sensor readings in metric semantic SLAM with the iSAM2 algorithm. We find inconclusive results, but detail future experiment directions in environments for perceptual ambiguity.

Index Terms—Autonomous Agents; Data Sets for SLAM; Multi-Robot SLAM; SLAM; Distributed Robot Systems

I. INTRODUCTION

Simultaneous localization and mapping (SLAM) is the process of building an understanding of an environment while also determining one's pose (position and orientation) within said environment. Robots typically perform active SLAM — moving around and interacting with the environment in question — to build their map. Map building is often accomplished by storing sensor measurements and landmarks in a data structure known as a pose graph. Pose graphs incorporate robot poses and environmental features (landmarks) as nodes connected by factors. Environments often contain features that make localizing within (and by extension mapping) them challenging. Unsafe scenarios occur when the SLAM algorithm fails because SLAM constitutes the robot's world perception. Consequences range from the minor case (*e.g.* a robot loses its way within a building) to catastrophic (*e.g.* an autonomous vehicle that fails to recognize a turn in the road).

A common phenomenon that induces failure is perceptual aliasing, where two different areas in the environment look identical through the robot's sensors. For example, two corridors that look similar, but are on separate floors of a building may induce perceptual aliasing and cause the robot to believe it is on a different floor (See Figure 2). Perceptual aliasing can cause incorrect data associations and result in erroneous constraints within the robot's pose graph, which affects the final map produced by the SLAM algorithm.

¹Thomas Snyder is with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, USA snyderth@oregonstate.edu

²Sudharshan Suresh is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA sudhars1@andrew.cmu.edu

³Michael Kaess is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA kaess@cmu.edu

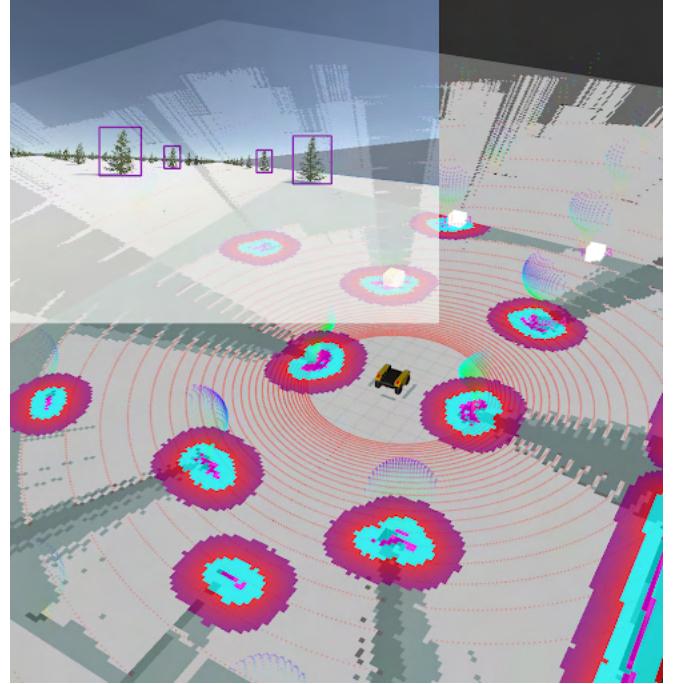


Fig. 1: The complete simulation user interface with the farm environment.

Failure in a single robot system often results in an incomplete task and requires an expensive, well-equipped robot.

Robot teams have become an interest within the SLAM literature. Multiple robots possess the ability to effectively understand an environment while reducing the workload of a single agent, requiring less computational power, and ultimately, less cost per robot. Centralized multiple robot SLAM systems use the team to gather and relay information to a single robot, which then performs optimization. Decentralized algorithms split the workload across the team, which improves robustness and reduces the computational workload of a single robot. Multiple robot systems are robust in that they may withstand hardware failure and still complete the task.

Despite utilizing multiple robots, perceptual aliasing remains an important problem to solve due to its ability to interfere with pose graph optimization via erroneous data constraints from false loop closures. Real world research into algorithm failure modes caused by perceptual aliasing introduces safety hazards and nondeterministic conditions. Unsafe conditions may result in expensive and time-consuming setbacks due to hardware damage or injury. Simulation creates



Fig. 2: (Top) The entrance to the bathrooms on the first floor of the learning and innovation center at Oregon State University. (Bottom) The entrance to the bathrooms on the second floor of the same building. The two corridors are identical in metric perception and are visually similar. This scenario is likely to cause erroneous data associations (perceptual aliasing).

a closed, safe environment to verify algorithms before testing in the real world. Real world is also limited by the options for variable environments. Environment reconfigurability is important for testing generalizable algorithms; simulation enables flexible dataset generation with less time and no added cost when compared with real-world experiments. For these reasons, we introduce three adversarial datasets that target perceptual aliasing to test existing metric semantic SLAM algorithms. These datasets may be easily extended to the multiple robot distributed SLAM case.

This paper is organized as follows: Section II details the distributed SLAM methods, SLAM datasets, and lack of datasets with perceptual aliasing. Section III details the three environments that we created. Section IV explains our results, and Section V discusses the implications of this work and future work.

II. RELATED WORK

Distributed SLAM enables multiple inexpensive robots to efficiently map an area while maintaining data privacy and independence from a powerful, centralized solver. Decentralizing SLAM requires incorporating multiple, sometimes inconsistent, measurements to assemble a coherent map of an area. Distributed data fusion smoothing and mapping 2.0 (DDF-SAM 2.0) was proposed to solve this problem [1]. The technique assembled local maps and periodically ran batch summarization to share maps with neighboring robots; data was fused through shared landmark observations. Choudhary et al. approached data fusion through a different pose graph formulation that incorporated inter-robot (robot-to-robot) and intra-robot (e.g. odometry) constraints [2]. Tian et al. proposed a distributed pose graph optimization (PGO) algorithm for collaborative SLAM based on Riemannian block coordinate descent, which is provably correct, but requires robot synchronization [3]. Tian et al. later removed the dependence on synchronization, which improved algorithmic robustness [4].

Distributed SLAM extends the classic SLAM front-end to incorporate factors from other robots. Wireless communication facilitates measurement dissemination between robots; although, operational scenarios often require lightweight, efficient wireless channel usage. Alongside their contribution to back-end fusion, Choudhary et al. incorporated high-level semantic representations in the front-end to address bandwidth constraints when sharing information among neighboring robots [2].

Robust distributed inference requires addressing algorithmic failure-modes. Perceptual aliasing is a well-known challenge, which often causes erroneous data associations that may induce failure-modes in existing active SLAM algorithms [5]. Methods commonly used to overcome this issue in single robot SLAM include maintaining multiple hypothesis on the environment [6], [7], explicitly modeling the aliasing [8], [9], or consistency checking against a trusted source of data [10]. Realizing, Reversing, and Recovering (RRR) verified loop closures by topologically partitioning the pose graph into “clusters,” which were checked for consistency against other clusters and the sequential constraints produced by robot odometry, which assumed odometry data was outlier-free [10]. Lajoie et al. tackled the odometry assumption by modeling the perceptual aliasing in the pose graph with discrete switching constraints [8]. Both of the previous examples synthetically incorporated perceptual aliasing via false data associations during experimentation.

Simulation is an important first step in algorithm verification, especially when testing algorithmic safety in challenging scenarios. Many algorithms benchmark performance on standard real world datasets in which erroneous constraints are artificially added [8], [10], [11]. Few progress to testing within photo-realistic 3D simulations, and even fewer implementations are on real robots [11], [12]. DOOR-SLAM tested a five-robot team using the ARGoS Simulator and a single, empty 2D environment and progressed to real world

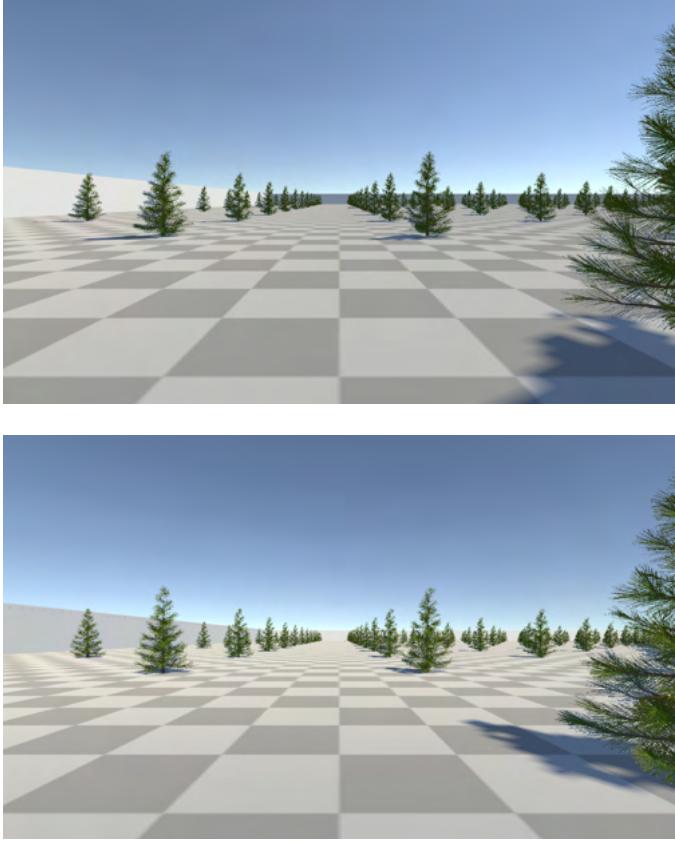


Fig. 3: (Top) The view facing the x-axis from the point (20, 100). (Bottom) The view facing the y-axis from the point (20, 20) meters. Notice the only visual difference in the shadow and lighting conditions.

robots [11]. Although DOOR-SLAM performed well in the real world, the 2D environment introduced limited perceptual ambiguity via sparse, similar features. The ARGoS map did not encompass other, more challenging, real world scenarios. For example, farming applications need to effectively navigate kilometers of uniform rows.

This paper builds on previous datasets, which were utilized in experiments involving GPS-denied scenarios. The previous environments included a military base [13], a hospital floor, and a bank [14]. More challenging scenarios that incorporate features, such as environments with high amounts of symmetry are necessary to verify algorithmic robustness in a safe environment. We implemented three simulation environments that incorporated semantic labels on objects and features likely to induce erroneous constraints via perceptual aliasing.

III. METHODS

We modeled three environments that may induce perceptual aliasing: a farm, a parking garage, and a forest. The environments were built using the Unity game engine coupled with ROS to simulate and command the robots and their stacks. For each dataset, we created a trajectory

based on velocity commands and performed ten trials. The inference algorithm was benchmarked via computed absolute trajectory error (ATE):

$$\text{ATE} = \left(\frac{1}{N} \sum_{i=0}^{N-1} \|\mathbf{X}_i - \hat{\mathbf{X}}_i\|^2 \right)^{\frac{1}{2}}, \quad (1)$$

where \mathbf{X} is ground truth trajectory and $\hat{\mathbf{X}}$ is estimated trajectory.

A. Environment I: Tree Farm

The first environment we tested was the field of a tree farm. Farms often possess large swaths of highly symmetric land with regularly spaced produce. The farm environment placed a two meter tall tree every 7.5 meters in a square pattern with 15 rows on a side. The first row was placed 7.5 meters from the origin. Walls bound the dataset on all four sides. Each wall is one meter thick with its center placed 7.5 meters away from the trees, totaling a square with 120 meter sides and walls placed at 120 meters from the origin. Robots were placed 20 meters from the corner of the arena. The regularly spaced trees and walls were meant to induce perceptual ambiguity via similar visual footprint (see Figure 3).

B. Environment II: Parking Garage

An empty parking garage was also selected because each floor is often identical to the previous floors (see Figure 2). In this case, the parking garage we used was a free asset from the Unity Asset store [15]. Each floor is 2.5 meters tall. The garage consists of four floors, excluding the roof, which may be accessed via a stairwell. Floors two and three are highly symmetric, with minor differences. Each floor has two ramps that provide access to the adjacent floors. The first and fourth floors are similar, but contain unique features, such as the parking garage entrance on the first floor and the lack of ramps leading to the floor above (See Figure 2). The Testing was not performed on this dataset due to the lack of sufficient support for Warthog z-axis navigation without manual intervention. Future work will extend the robot navigation stack to incorporate the necessary intelligence for traversing inclines with the express purpose of attaining a goal in an elevated area.

C. Environment III: Forest

The final dataset is an old-growth forest, with five thousand trees randomly dispersed throughout the one kilometer square environment. Previous attempts with ten thousand trees proved to be too dense for the robot's planner. Although tree placement is random, types of trees are not distinguishable to the robot (all labels are simply "tree"), and odometry drift coupled with the large number of trees increases the likelihood of inducing erroneous loop closures in the underlying pose graph. The forest environment is a highly applicable scenario with a large body of single robot work, yet distributed inference has not to our knowledge begun testing on such datasets [16].

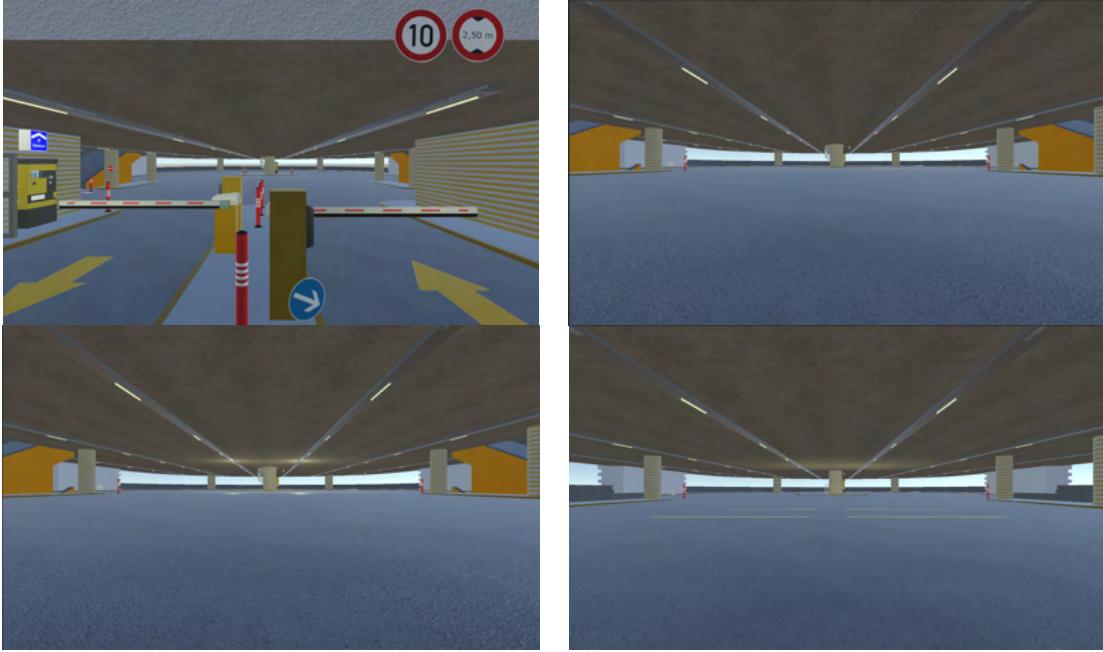


Fig. 4: (Top left) The first floor of the parking garage, taken from the entrance. (Top right) Second floor of the parking garage. (Bottom left) Third floor of the parking garage. (Bottom right) Fourth floor of the parking garage. Notice that the floor layout on floors two and three is identical, causing similar metric and visual measurements. The fourth floor lacks the ramps that access the upper floor, and the first floor includes the entrance (the view of the back wall is behind the camera in the second, third, and fourth floors, but is identical for those three floors).

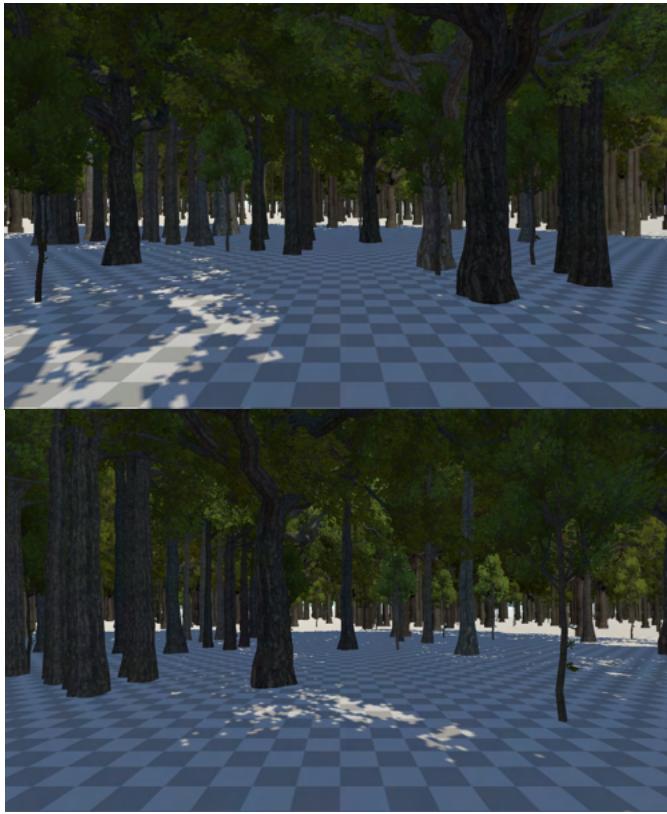


Fig. 5: Two views from the forest dataset that are 100 meters apart in the x-axis.

Environment	ATE Average (m)	ATE Std. (m)
Farm	39.41	23.76
Forest	18.45	6.56

TABLE I: Absolute trajectory error for farm and forest datasets averaged over all ten trials.

Parameter	Value
Environment	Farm/Forest/Parking Garage
Robot	Clearpath Robotics Warthog
Sensing	IMU, LIDAR, Stereo camera, Monocular camera
Max Velocity	2 m/s
Max Rotational Velocity	2.5 rad/s
Max Acceleration	20 m/s ²
Inference Algorithm	iSAM2
Trials	10

TABLE II: Summary of experimental parameters.

IV. RESULTS

Experiments were performed with Clearpath Robotics Warthogs as the robotic platform due to their rich pre-existing sensor suite. The robots possessed a LIDAR, single forward-facing camera intended for object detection, and a forward-facing stereo imaging system for depth information. The robot's top speed was limited to 2 m/s, its rotational speed was limited to 2.5 rad/s, and its acceleration was limited to 20 m/s². Experimental constants are shown in Table II.

A. Qualitative Results

The robots were typically unable to complete the trials without manual intervention. The planner often failed to converge on a solution. Failure to converge usually occurred when the robot was in close proximity to an object (*e.g.* after a head-on collision); however, there were some cases where the planner failed to complete the trajectory given ample space. In the latter scenario, the robot's planner was reinitialized and it always completed the trajectory. When the robot collided head-on with an obstacle, the robot was always unable to re-plan, which ended the trial. The robot was less likely to complete the farm trajectory than the forest trajectory because the robot tended to collide with objects more often in the farm dataset. This tendency is most likely due to the amount of spacing between the trees; however, there were scenarios in the forest environment when the robot was able to navigate smaller spaces.

B. Quantitative Results

The robot performed poorly on all datasets with 39.41 m average ATE on the farm dataset and 18.45 m average ATE on the forest dataset, excluding outliers. A trial was considered an outlier if the ATE was greater than two standard deviations from the mean when included in the computations. One trial was an outlier on the forest dataset and there were zero outliers on the farm dataset. The metrics are summarized in Table I.

Notably, the performance varied significantly between trials, which was captured by the ATE standard deviation. The wide variance may be attributed to the robot's freedom to plan its path between waypoints, which may have changed the portion of the environment that the robot sensed. The robot's freedom does not explain the marked difference between the ATE across the two environments. The difference reflects the robot's success in completing each dataset. The robot completed the farm dataset less than the forest dataset, which meant that the SLAM front end found less or no loop closures in the farm when compared to the forest. The farm dataset's trajectory was also longer than the forest dataset's trajectory, which contributed to increased odometry drift without correction. This effectively tested the planning algorithm, but the experiment failed to provide meaningful results as to whether these environments induced perceptual aliasing in the SLAM front end.

V. DISCUSSION AND CONCLUSION

This paper contributes three adversarial environments for SLAM algorithms that were expected to produce scenarios that increase the probability of perceptual aliasing. The environments are commonly found in real world scenarios, which facilitates a smooth transition from simulation to the real world. We showed two results in which a single robot attempting to build a map of the environment using the iSAM2 inference framework failed to do so because of difficulty in planning around the dataset, which often meant that loop closures were not present in the data.

Future work will tune the planner for these environments or change the environments, such that the planner is able to navigate without faults. Future work also includes incorporating a framework that entertains multiple hypotheses for a new data association to overcome perceptual aliasing in the backend. These datasets are extensible to multiple robot scenarios, allowing their use in distributed SLAM frameworks, ultimately assisting the development of robust and safe distributed inference algorithms.

ACKNOWLEDGMENT

This work was funded by a National Science Foundation REU grant. The author would like to thank Ms. Rachel Burcin and Dr. John M. Dolan for their support and dedication to the RISS program and its scholar's success. The author would also like to thank Akash Sharma, Dan McGann, and Wei Dong from the Robot Perception Lab for their support and guidance along the way.

REFERENCES

- [1] A. Cunningham, V. Indelman, and F. Dellaert, "DDF-SAM 2.0: Consistent distributed smoothing and mapping," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2013, pp. 5220–5227.
- [2] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. I. Christensen, and F. Dellaert, "Distributed Mapping with Privacy and Communication Constraints: Lightweight Algorithms and Object-based Models," *Journal of Vibration and Control*, p. 107754631982824, feb 2017. [Online]. Available: <https://arxiv.org/pdf/1702.03435.pdf>
- [3] Y. Tian, K. Khosoussi, D. M. Rosen, and J. P. How, "Distributed Certifiably Correct Pose-Graph Optimization," *IEEE Transactions on Robotics*, nov 2019. [Online]. Available: <http://arxiv.org/abs/1911.03721>
- [4] Y. Tian, A. Koppel, A. S. Bedi, and J. P. How, "Asynchronous and Parallel Distributed Pose Graph Optimization," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5819–5826, oct 2020. [Online]. Available: <https://ieeexplore.ieee.org/document/9143442/>
- [5] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, p. 1309–1332, Dec 2016. [Online]. Available: <http://dx.doi.org/10.1109/TRO.2016.2624754>
- [6] M. Hsiao, J. G. Mangelson, S. Suresh, C. Debrunner, and M. Kaess, "ARAS: Ambiguity-aware Robust Active SLAM based on Multi-hypothesis State and Map Estimations," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, oct 2020, pp. 5037–5044. [Online]. Available: <https://ieeexplore.ieee.org/document/9341384/>
- [7] F. Jiang, V. Agrawal, R. Buchanan, M. Fallon, and F. Dellaert, "iMHS: An incremental multi-hypothesis smoother," 2021.
- [8] P. Y. Lajoie, S. Hu, G. Beltrame, and L. Carlone, "Modeling Perceptual Aliasing in SLAM via Discrete-Continuous Graphical Models," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1232–1239, 2019.
- [9] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020. [Online]. Available: <https://github.com/MIT-SPARK/Kimera>
- [10] Y. Latif, C. Cadena, and J. Neira, "Robust loop closing over time for pose graph SLAM," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1611–1626, oct 2013. [Online]. Available: <https://doi.org/10.1177/0278364913498910>
- [11] P. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "DOOR-SLAM: Distributed, Online, and Outlier Resilient SLAM for Robotic Teams," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1656–1663, 2020.
- [12] Y. Chang, Y. Tian, J. P. How, and L. Carlone, "Kimera-multi: a system for distributed multi-robot metric-semantic simultaneous localization and mapping," 2020.

- [13] J. G. Rogers, J. R. Fink, and E. A. Stump, "Mapping with a ground robot in GPS denied and degraded environments," in *2014 American Control Conference*, 2014, pp. 1880–1885.
- [14] N. Fung, J. Rogers, C. Nieto, H. I. Christensen, S. Kemna, and G. Sukhatme, "Coordinating multi-robot systems through environment partitioning for adaptive informative sampling," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 3231–3237.
- [15] 255 pixel studios, "Parking garage - Complete: 3D urban: Unity asset store." [Online]. Available: <https://assetstore.unity.com/packages/3d/environments/urban/parking-garage-complete-104909>
- [16] A. S. Aguiar, F. N. dos Santos, J. B. Cunha, H. Sobreira, and A. J. Sousa, "Localization and Mapping for Robots in Agriculture and Forestry: A Survey," *Robotics*, vol. 9, no. 4, p. 97, nov 2020. [Online]. Available: <https://www.mdpi.com/2218-6581/9/4/97>
- [17] T. Cieslewski, S. Choudhary, and D. Scaramuzza, "Data-efficient decentralized visual SLAM," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2018, pp. 2466–2473.
- [18] R. Sharma, C. Taylor, D. Casbeer, and R. Beard, "Distributed Cooperative SLAM using an Information Consensus Filter*," in *AIAA Guidance, Navigation, and Control Conference*, no. June 2017. Reston, Virginia: American Institute of Aeronautics and Astronautics, aug 2010. [Online]. Available: <https://arc.aiaa.org/doi/10.2514/6.2010-8334>
- [19] M. Montemerlo, S. Thrun, D. Roller, and B. Wegbreit, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *IJCAI International Joint Conference on Artificial Intelligence*, 2003, pp. 1151–1156.

Concept Whitening for Transfer of Concepts in Imitation Learning

Grace Su¹, Ini Oguntola², Katia Sycara², Dana Hughes², Joseph Campbell²

Abstract—Robots and other autonomous agents often require large amounts of training data in order to accomplish specifically-defined tasks. At the same time, these agents are limited to certain environments and their performances are negatively impacted by domain shifts. Concept whitening is an interpretable machine learning mechanism proposed by Chen et al. [1] that aligns the latent space of a neural network’s layer with concepts of interest without hurting performance. Though concept whitening was initially proposed for image recognition, it is a generalizable deep interpretability technique that has the potential to be applied to other neural networks to improve the interpretability of the model. However, in the current literature, it is unknown whether such an interpretability technique can be used as a general method to transfer learned concepts between similar tasks. In the present paper, we use concept whitening on single-agent, recurrent actor-critic imitation learning models for tasks in a simple 2D gridworld to investigate whether learned relationships between gridworld objects help reduce the amount of training data needed in order to train the same models on similar tasks. The results show that in recurrent imitation learning, concept whitening can perform a small amount of latent space alignment towards defined concepts but applying concept whitening to this task is not trivial.

Index Terms—Concept Whitening, Interpretability, Transfer Learning, Imitation Learning, Deep Learning

I. INTRODUCTION

In order for autonomous agents to be successfully deployed in the real world, these agents should be able to interact with humans and accomplish their goals despite ever-changing environmental conditions. Thus, one of the most important challenges in developing autonomous agents is developing adaptive, robust, and intelligent systems. In addition to these qualities, these systems must also be interpretable to facilitate human-agent interaction. However, these characteristics are difficult to obtain. This is especially true in deep reinforcement learning, a method that can allow agents to achieve high performance on tasks but requires large amounts of data. The resultant policy is only applicable to tasks and environments that are similar to those that were also in the training data [2]. Thus, transfer learning, where learned parameters are shared with a new model that is then fine-tuned and trained further, can improve the new model’s generalization and learning efficiency [3]. In the current literature, a number of transfer learning techniques have been proposed, but few attempt to utilize interpretable components of trained models [2].

¹Grace Su is with the Computer Science Department, Columbia University, New York, NY., USA. g.su@columbia.edu

²Ini Oguntola, Katia Sycara, Dana Hughes, and Joseph Campbell are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA., USA. {ioguntola, sycara, danahugh, jcampbell}@andrew.cmu.edu

Agents that interact with humans need to be interpretable. However, a search-and-rescue agent, compared to humans, need lots of demonstration data to navigate a certain type of environment. The agent does not perform well on environments with slightly different parameters, rules, or objectives. In addition, the agent does not recognize which rules are shared among different environments.

In this paper, we investigate if using interpretable components of trained models, specifically layers trained with concept whitening, can facilitate transfer learning. Section II gives a background on related work and what strengths and weaknesses concept whitening has as an interpretability technique. Section III discusses the specific machine learning algorithms we use to train a simple autonomous agent. Section IV details the experiments performed to evaluate model interpretability and transfer learning. Section V presents and analyzes experimental results. Finally, we conclude the paper and describe future work and potential applications.

II. RELATED WORK

A. Imitation Learning

Imitation learning is a method in which the autonomous agent extracts useful skills and learns to replicate behaviors from expert demonstrations [4]. It has demonstrated strong performance in areas such as object manipulation, video games, and autonomous driving. But, the performance of agents trained with imitation learning are limited by the quality and scope of the demonstrations the agents were trained on. Some imitation learning methods employ techniques to make the models more interpretable. One approach uses human-readable programmatic policies instead of black-box networks [5]. Another approach, InfoGAIL, uses information theoretic losses to infer the latent structure of demonstrations and therefore reveal semantically meaningful data variations [6]. However, most imitation learning methods do not necessarily consider interpretability.

B. Transfer Learning

Transfer learning methods help solve problems in which there is a lack of data from the target domain, like robotics. Within robotics, three approaches to transfer learning emerge: better simulation, policy randomization, and robust policy [2]. The first approach is improving the simulation environments in which data is generated so that behavior learned from the simulation is transferred more effectively to an agent operating in the real world. This method provides an abundant source of data but is prone to modeling errors. A second approach is policy randomization, which can produce highly adaptive strategies, reduce reliance

upon modeling, and therefore aid transfer from simulation to the real world [7]–[9]. A final approach is learning a robust policy that learns skills important for generalization in addition to completing tasks [10], [11].

C. Deep Interpretability

In order to make deep learning models more interpretable, post-hoc methods provide interpretability after the model is trained, and are thus easily applicable to many different models. Concept-vector methods attempt to provide explanations based on higher-level concepts, but make assumptions about the model’s latent space that may not hold [12]. Meanwhile, saliency methods highlight the most important input features, but these features tend to be low-level and not as human-interpretable.

There are also methods with models that are interpretable by design so the latent space is aligned with human-interpretable concepts. For instance, concept whitening in image classification led model activations to be aligned with different image classes [1]. Concept whitening has also been applied to theory of mind models for modeling other entities’ intentions, which not only increased interpretability, but also increased predictive performance [12]. This increase in performance causes reason to suspect that aspects of a model trained with concept whitening could be effectively used for transfer learning in autonomous agents.

III. METHODS

We chose to train imitation learning models with concept whitening because the demonstrations can be used to generate concept datasets, which are necessary for concept whitening. Concept whitening is the most useful when the concepts used potentially have information that is not immediately obvious from the model input data.

Thus, to obtain suitable, simple gridworld environments and tasks for single-agent deep reinforcement learning models, we used the BabyAI research platform [13]. The platform is “an extensible suite of 19 levels of increasing difficulty,” in which each level is an instruction-following task.

A. Imitation Learning Model

The BabyAI platform includes an imitation learning model. The model architecture is shown in the appendix in Figure 6. It is a recurrent actor-critic model with the objective of achieving behavioral cloning from the expert demonstrations provided as training data. The GRU (Gated Recurrent Unit) [14] encodes the language input instruction. Then, two batch-normalized FiLM (Feature-wise Linear Modulation) [15] layers and a convolutional network jointly processes the observation and instruction. At each step, a LSTM (Long Short-Term Memory) cell [16] temporally integrates representations produced by the FiLM layers at each step. The memory has 128 units and encodes the instruction with a unidirectional GRU and no attention mechanism because the task domain we chose is relatively simple. The model uses the Adam optimizer [17] with the hyperparameters used by the BabyAI research platform.

B. Concept Whitening

The original concept whitening module developed by Chen et al. [1] was for use in image classification models. Imitation learning, in comparison, is a significantly different machine learning task because the environment is sequential and only partially observable while image classification is episodic and fully observable. However, we can apply concept whitening to imitation learning because the technique is model-agnostic and has been successfully implemented in theory of mind models for inferring intents of other agents [12].

The concept whitening mechanism trains a neural network layer’s latent space to align with an explicitly-defined set of concepts. To implement this, a concept whitening module can be substituted for a normalization module, such as batch normalization. Then, the module requires datasets of the desired concepts in order to learn an orthogonal matrix that aligns the latent space with those concepts. To incorporate concept whitening in the BabyAI imitation learning model, we replaced the first batch normalization layer with a concept whitening layer as shown in the appendix, Figure 7. Replacing the first batch normalization layer led to more concept alignment than replacing the second batch normalization layer.

During the forward pass of the overall model’s training algorithm, we use the same forward pass algorithm from the original concept whitening implementation [1]:

Algorithm 1 Forward Pass of CW Module

Input: mini-batch input $\mathbf{Z} \in \mathbb{R}^{d \times m}$

Optimization Variables: orthogonal matrix $\mathbf{Q}_{d \times d}$ (learned in Algorithm 2)

Output: whitened representation $\hat{\mathbf{Z}} \in \mathbb{R}^{d \times m}$ (learned in Algorithm 2)

- 1: calculate batch mean: $\mu = \frac{1}{m}\mathbf{Z} \cdot \mathbf{1}$, and center the activation: $\mathbf{Z}_C = \mathbf{Z} - \mu \cdot \mathbf{1}^T$
 - 2: calculate ZCA-whitening matrix \mathbf{W} , for details see Algorithm 1 of [18]
 - 3: calculate the whitened representation: $\hat{\mathbf{Z}} = \mathbf{Q}^T \mathbf{W} \mathbf{Z}_C$
-

As Chen et al. [1] explains, “ $\mathbf{Z}_{d \times n}$ is the latent representation matrix of n samples, in which each column $\mathbf{z}_i \in \mathbb{R}^d$ contains the latent features of the i^{th} sample.” $\mathbf{Q}_{d \times d}$ is the orthogonal matrix whose column \mathbf{q}_j is the j^{th} axis and is learned during optimization of the concept whitening module.

C. Training

We replaced the first batch normalization layer with a concept whitening layer as shown in the appendix, Figure 15. Then, model training alternates between optimizing for main objective function of behavioral cloning of expert, in order to succeed in level missions, and optimization of the concept whitening objective. The concept whitening objective updates the orthogonal matrix \mathbf{Q} with concepts by maximizing the matrix’s activation along axis j for each concept j ’s data [1].

The model optimization algorithm we use is summarized in Algorithm 1. It is similar to Algorithm 2 used in [1], but because there is an LSTM cell before the concept whitening layer that affects the input to the concept whitening layer, memory data corresponding to each concept data batch must also be used to update the concept whitening layer.

Algorithm 2 Recurrent Actor-Critic Concept Whitening Training

```

Input: main objective dataset  $D$ , concept datasets
 $\mathbf{X}_{c_1}, \mathbf{X}_{c_2}, \dots, \mathbf{X}_{c_k}$ 
Parameters:  $\beta =$  number of concept batches to use during
training
1: for epoch 1 to  $N$  do
2:   Complete one epoch of training for the model's main
objective function using batches sampled from  $D$ 
3:   for  $r = 0$  to  $\beta$  do
4:     for  $j = 1, \dots, k$  do
5:       Load one random batch  $b$  from concept dataset
 $\mathbf{X}_{c_j}$ 
6:       Load a batch  $P$  containing each full demonstration
the concepts in batch  $b$  belong to
7:       memory  $m \leftarrow$  pass  $P$  through the model
8:        $m_{concept} \leftarrow m$  at timesteps in  $P$  corresponding
to data in  $b$ 
9:       Using concept data inputs  $m_{concept}$  and  $b$ , calculate
 $\mathbf{G} = \nabla_{\mathbf{Q}}$ 
10:      Update matrix  $\mathbf{Q}$  using  $\mathbf{G}$ . For details see Algo-
rithm 2 of [1]
11:    end for
12:   end for
13: end for
```

IV. EXPERIMENTS

A. Task Domain

Initially, we chose the BabyAI [13] “GoToImpUnlock” level as our task domain. Missions given to the agent are of the form “Go to the [color] [object]”. The agent must navigate a a 3x3 maze of 6x6 rooms, randomly inter-connected by doors and unlock doors when necessary. Colored doors are unlocked by keys with the matching color. An example of a randomly generated level is shown in Figure [].

“GoToImpUnlock” is considered one of the more difficult levels in BabyAI because it requires the agent to learn many different competencies. It includes basic competencies such as going to a specified object, navigating a maze with multiple rooms, and ignoring distractor objects. At the same time, “GoToImpUnlock” requires the agent to determine if the mission instructions implicitly require a door to be unlocked with a key. The benefit of using such a relatively difficult BabyAI is that understanding the behavior of the agent is nontrivial, so concept whitening would likely provide a helpful improvement in model interpretability. However, after training several concept whitening models with different hyperparameters on the “GoToImpUnlock”

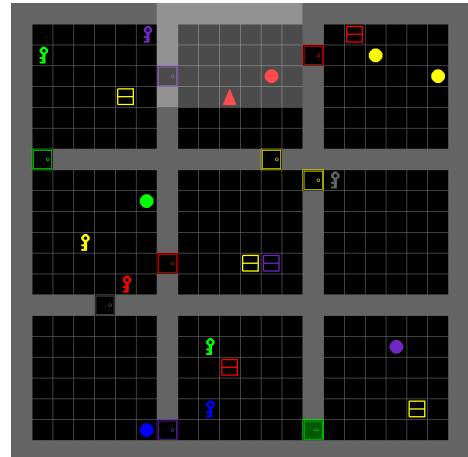


Fig. 1: Example BabyAI “GoToImpUnlock” level with mission “Go to the purple ball”. The agent and the direction it is currently facing is represented by the red triangle. The highlighted area around the agent represents the environment tiles currently visible to the agent.

level, we found that the concept whitening layer did not seem to be effecting any significant changes on the model’s latent space. The concept whitening layer’s mean activations for each concept showed no significant differences when the concept of the input data changed. The mean activation plots and significance are further discussed in Section V: Results.

Thus, using the BabyAI research platform, we created our own “UnlockRGB” level as our task domain. It is a simple environment that still encourages the model to learn latent concepts in the input data to perform better at the main imitation learning task. Missions are of the form ”Open the [color] door”. [Color] is limited to either “red”, “green”, or “blue”. The environment consists of 2 rooms connected by a colored door, which always must be unlocked the key of the matching color. An example of a randomly generated level is shown in Figure 2.

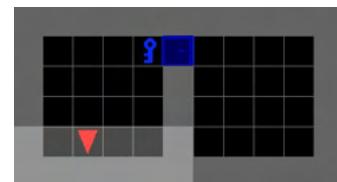


Fig. 2: Example “UnlockRGB” level with mission “Open the blue door”. The agent and the direction it is currently facing is represented by the red triangle. The highlighted area around the agent represents the environment tiles currently visible to the agent.

B. Demonstrations

Demonstrations are generated by the hand-crafted bot agent provided by BabyAI. In order to perform the role of a simulated human teacher, it parses the mission instructions and environment parameters, then creates and maintains

a stack of subgoals to complete the given mission. The subgoals which the bot implements are:

- **Open:** Open a door that is in front of the agent.
- **Close:** Close a door that is in front of the agent.
- **Pickup:** Execute the pickup action (pick up an object).
- **Drop:** Execute the drop action (drop an object being carried).
- **GoNextTo:** Go next to an object matching a given (type, color) description or next to a cell at a given position.

A demonstration consists of a list of (observation, action, done) Observations are partial and egocentric. The agent observes a square of 7x7 tiles in the direction agent is facing, includes the tile the agent is standing on, but at the same time the agent cannot see through walls or closed doors. Additional input: Mission in the form of a simple language input. Following the precedent set by the BabyAI paper, we generate a set of 1 million demonstrations [13].

C. Concepts

We designed “UnlockRGB” to help the agent learn that the key color must match the door color in order for unlocking to succeed. This color relationship between objects can be learned through processing of the demonstration data, but it is still a latent input data feature that would not be revealed unless the model is made more interpretable through concept whitening.

The concept datasets are created by labeling demonstration data at each timestep with the corresponding subgoal used by the bot at that timestep. For example: a demonstration with a mission asking the agent to “Open the red door” would have the timesteps from when the bot first moves towards the key with the “GoNextTo” subgoal, to when the bot picks up the key, labeled as concept “0. Search for red key”. Then, the timesteps from when the bot picks up the key, to when the bot performs the unlocking action on the door, are labeled as concept “3. Take red key to door”.

For demonstrations asking the agent to “Open the green door”, the demonstration data is labeled with concepts at the correct timesteps using the same method, but the labels are “1. Search for green key” and “4. Take green key to door” instead. Similarly, for demonstrations asking the agent to “Open the blue door”, the data is labeled using the same method, but the labels are “1. Search for blue key” and “4. Take blue key to door” instead. The set of concept labels used are shown in Figure 3:

0. Search for red key

1. Search for green key
2. Search for blue key
3. Take red key to door
4. Take green key to door
5. Take blue key to door

Fig. 3: “UnlockRGB” concept labels, color-coded by the mission door color

V. RESULTS

A. Training Metrics

After developing a new level tailored for evaluating concept whitening in imitation learning on the BabyAI platform, and generating the level’s training demonstrations and concept datasets, we show that the baseline model training results are the same as the model trained with concept whitening, as expected. The training metrics entropy, policy loss, training accuracy, validation accuracy, validation return, and validation success rate remain approximately the same across all training timesteps, as shown in Figure 4, while the concept whitening additionally aligns the latent layer to be more interpretable.

This demonstrates we have been able to create an implementation and application of concept whitening for a recurrent actor-critic imitation learning model. Our developed framework for training the concept whitening model and analyzing the concept whitening layer’s activations can be used in future work.

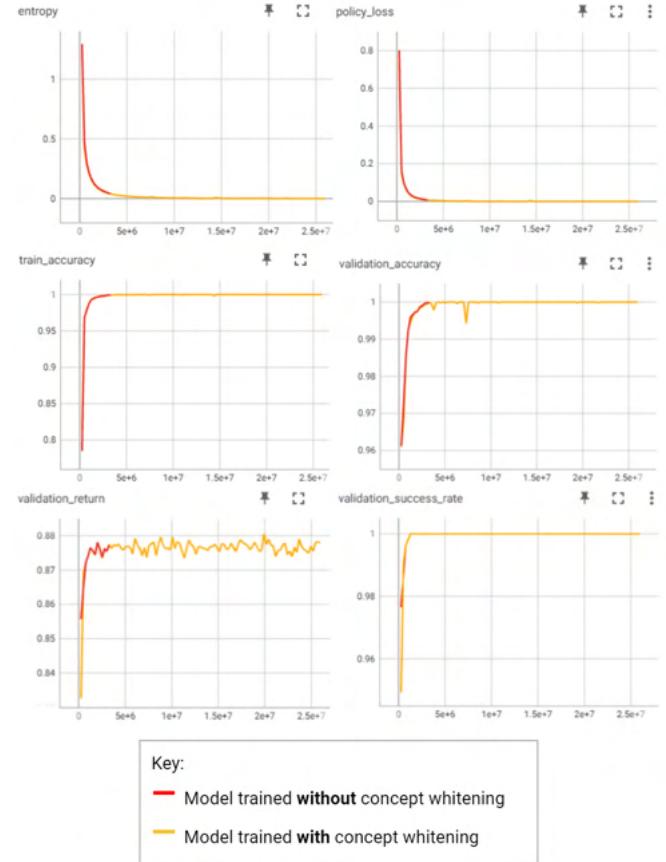


Fig. 4: Training metrics across all training timesteps

B. Mean Concept Activations and Analysis

The objective of concept whitening is to maximize the activation of a given concept’s data input on the corresponding concept column in the orthogonal matrix \mathbf{Q} [1]. Thus, we ideally expect a concept’s mean activations to be highest

when the model receives that concept as input. The activation for each column j of \mathbf{Q} is given by

$$a_j = \mathbf{q}_j^T \hat{\mathbf{z}}_j \quad (1)$$

where $\hat{\mathbf{z}}_j$ is the concept-whitened latent representation for one demonstration data input belonging to concept j .

To conduct some preliminary analysis, we sample 300 different demonstrations from each concept dataset. The resulting number of samples belonging to each concept are shown in Table 1:

TABLE I: Number of Samples Belonging to Each Concept

Concept 0	9692
Concept 1	9531
Concept 2	9703
Concept 3	11757
Concept 4	11579
Concept 5	11667

Then, we calculate the mean activations of the concept whitening layer for each concept’s samples, and visualize them as bar plots in Figure 5.

Analyzing the bar plots reveals that the mean activation on a concept column has a small relative increase when its corresponding concept data is passed through the concept whitening layer. At the same time, there also appears to be a small relative increase for the other concept column with the same color. This suggests that the model is learning that there is some positive correlation between searching for a key of a specific color and taking that key of the same color to the door, which is what we would expect in order for the model to learn how to unlock colored doors.

To measure how effective the concept whitening was at aligning the latent space axes with each of the concepts, we also train a decision tree classifier to predict which concept data input generated each activation vector. The decision tree achieves an accuracy of 84.5 percent, which is better than random.

These quantitative assessments show that the concept whitening layer is able to align the latent space with the chosen concepts to a small degree. But overall, compared to concept whitening results from [1] and [12], the recurrent model type and imitation learning task appear to decrease the potency of the concept whitening layer’s alignment procedure. This suggests that using concept whitening in this recurrent imitation learning task is not a trivial matter of replacing one of the model’s batch normalization layers with concept whitening and tuning the hyperparameters.

C. Hyperparameters

The hyperparameters of the concept whitening were determined through multiple trials of training. We find that iteration frequency $T = 8$ and momentum $p = 0.5$ facilitate separation between mean concept activation values the most. There were four different calculations proposed by [1] to define activation based on the feature map: (a) mean of all

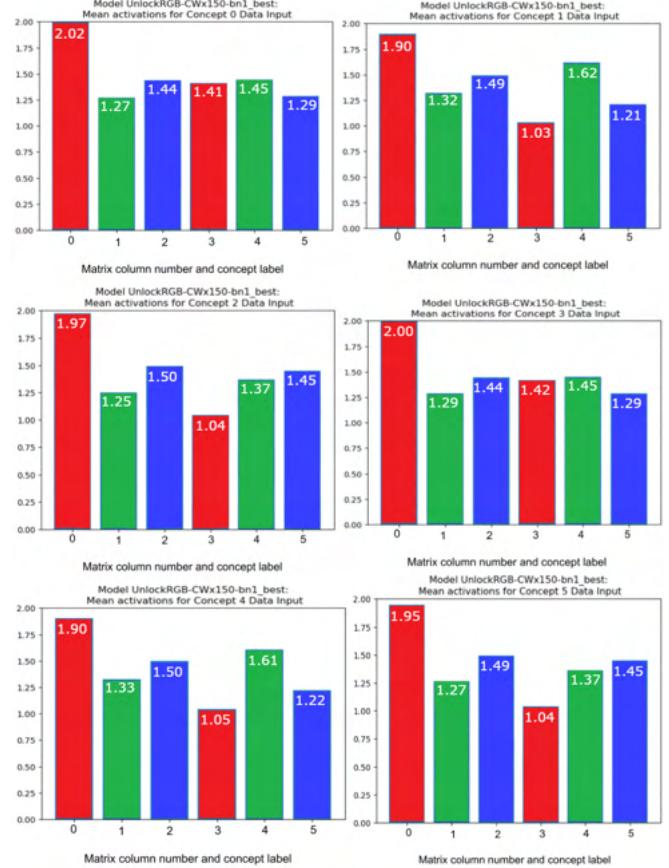


Fig. 5: Plots of mean activations for data inputs from concepts 0 through 5, in that order. Each bar represents the mean activation of each concept column in matrix \mathbf{Q} and has its value written in white text. Bars are labeled on the horizontal axis with the concept columns they correspond to.

feature map values; (b) max of all feature map values; (c) mean of all positive feature map values; (d) mean of down-sampled feature map obtained by max pooling. After testing all activations, we found that (a) taking the mean of all feature map values allowed the concept whitened model to achieve the closest performance to the baseline model. This is likely because our imitation learning task there is no spatial encoding of the input observation data required to the degree necessary in image classification.

For $\beta =$ the number of concept batches to use during training as defined in Algorithm 2, we found that $\beta = 150$ facilitates separation between mean concept activation values the most. This is because the number of batches used from the main objective dataset during training was 100. If β is not at least as large as the number of main objective dataset batches, the concept whitening updates are not strong enough to consistently increase the activation values of a given concept’s data on the corresponding concept column of \mathbf{Q} .

VI. FUTURE WORK

The results suggest that concept-whitening in the recurrent imitation learning context is able to align the latent space with the chosen concepts to a small degree, but it is non-trivial to implement and requires additional considerations specific to the learning task. Thus, the next steps include testing different concept definitions and strengthening the concept whitening updates.

To strengthen the concept whitening updates, different models may be considered. For instance, the recurrence in our model caused the concept whitening to require concept data memories in addition to labeled demonstration timesteps. The memory data contains information about the entire demonstration up to the timestep of the concept data batch, so removing recurrence could help ensure the concept data inputs, and therefore the resulting activations, remain separate from each other.

Additionally, the behavioral cloning objective of our model may not require learning about different concepts, so then concept whitening layer would not have latent space data with encoded concepts to align towards. In that case, a reinforcement learning-only model may be desirable. Different levels with concepts certain to be latent in the main objective could provide additional assistance to the concept whitening layer.

Obtaining concept whitening layers with strongly-aligned concept activations can then help us better investigate whether transferring a trained concept whitening layer to levels that use the same concepts, but have somewhat different objectives, improves data efficiency. Overall, these steps will allow us to discover whether using whitened concept representations produced by a model trained on one task domain A helps reduce the amount of training data needed or improve performance on a similar, but different task domain B.

ACKNOWLEDGMENT

This material is based upon work supported by the National Science Foundation Research Experience for Undergraduates Program under Grant No. 1659774. Thank you to the mentors, sponsors, the RISS team, and CMU for making this opportunity possible.

REFERENCES

- [1] Z. Chen, Y. Bei, and C. Rudin, “Concept whitening for interpretable image recognition,” *Nature Machine Intelligence*, vol. 2, no. 12, p. 772–782, Dec 2020. [Online]. Available: <http://dx.doi.org/10.1038/s42256-020-00265-z>
- [2] J. Hua, L. Zeng, G. Li, and Z. Ju, “Learning for a robot: Deep reinforcement learning, imitation learning, transfer learning,” *Sensors*, vol. 21, no. 4, 2021. [Online]. Available: <https://www.mdpi.com/1424-8220/21/4/1278>
- [3] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine, “Learning invariant feature spaces to transfer skills with reinforcement learning,” 2017.
- [4] B. Zheng, S. Verma, J. Zhou, I. W. Tsang, and F. Chen, “Imitation learning: Progress, taxonomies and opportunities,” *CoRR*, vol. abs/2106.12177, 2021. [Online]. Available: <https://arxiv.org/abs/2106.12177>
- [5] A. Verma, V. Murali, R. Singh, P. Kohli, and S. Chaudhuri, “Programmatically interpretable reinforcement learning,” *CoRR*, vol. abs/1804.02477, 2018. [Online]. Available: <http://arxiv.org/abs/1804.02477>
- [6] Y. Li, J. Song, and S. Ermon, “Infogail: Interpretable imitation learning from visual demonstrations,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS’17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 3815–3825.
- [7] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, “Deepmimic: Example-guided deep reinforcement learning of physics-based character skills,” *ACM Trans. Graph.*, vol. 37, no. 4, July 2018. [Online]. Available: <https://doi.org/10.1145/3197517.3201311>
- [8] H. B. Ammar, E. Eaton, P. Ruvolo, and M. E. Taylor, “Online multi-task learning for policy gradient methods,” in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ser. ICML’14. JMLR.org, 2014, p. II–1206–II–1214.
- [9] A. A. Rusu, M. Večerík, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, “Sim-to-real robot learning from pixels with progressive nets,” in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 13–15 Nov 2017, pp. 262–270. [Online]. Available: <http://proceedings.mlr.press/v78/rusu17a.html>
- [10] R. Ramakrishnan, E. Kamar, D. Dey, E. Horvitz, and J. Shah, “Blind spot detection for safe sim-to-real transfer,” *J. Artif. Intell. Res.*, vol. 67, pp. 191–234, 2020. [Online]. Available: <https://doi.org/10.1613/jair.1.11436>
- [11] Z. He, R. Julian, E. Heiden, H. Zhang, S. Schaal, J. J. Lim, G. S. Sukhatme, and K. Hausman, “Zero-shot skill composition and simulation-to-real transfer by learning task representations,” *CoRR*, vol. abs/1810.02422, 2018. [Online]. Available: <http://arxiv.org/abs/1810.02422>
- [12] I. Oguntola, D. Hughes, and K. Sycara, “Deep interpretable models of theory of mind,” 2021.
- [13] M. Chevalier-Boisvert, D. Bahdanau, S. Lahoud, L. Willem, C. Saharia, T. H. Nguyen, and Y. Bengio, “BabyAI: First steps towards grounded language learning with a human in the loop,” in *International Conference on Learning Representations*, 2019. [Online]. Available: <https://openreview.net/forum?id=JcXCo0cYX>
- [14] K. Cho, B. van Merriënboer, Ç. Gülcehre, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” *CoRR*, vol. abs/1406.1078, 2014. [Online]. Available: <http://arxiv.org/abs/1406.1078>
- [15] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, “Film: Visual reasoning with a general conditioning layer,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.
- [16] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, p. 1735–1780, Nov. 1997. [Online]. Available: <https://doi.org/10.1162/neco.1997.9.8.1735>
- [17] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [18] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261–2269.

APPENDIX

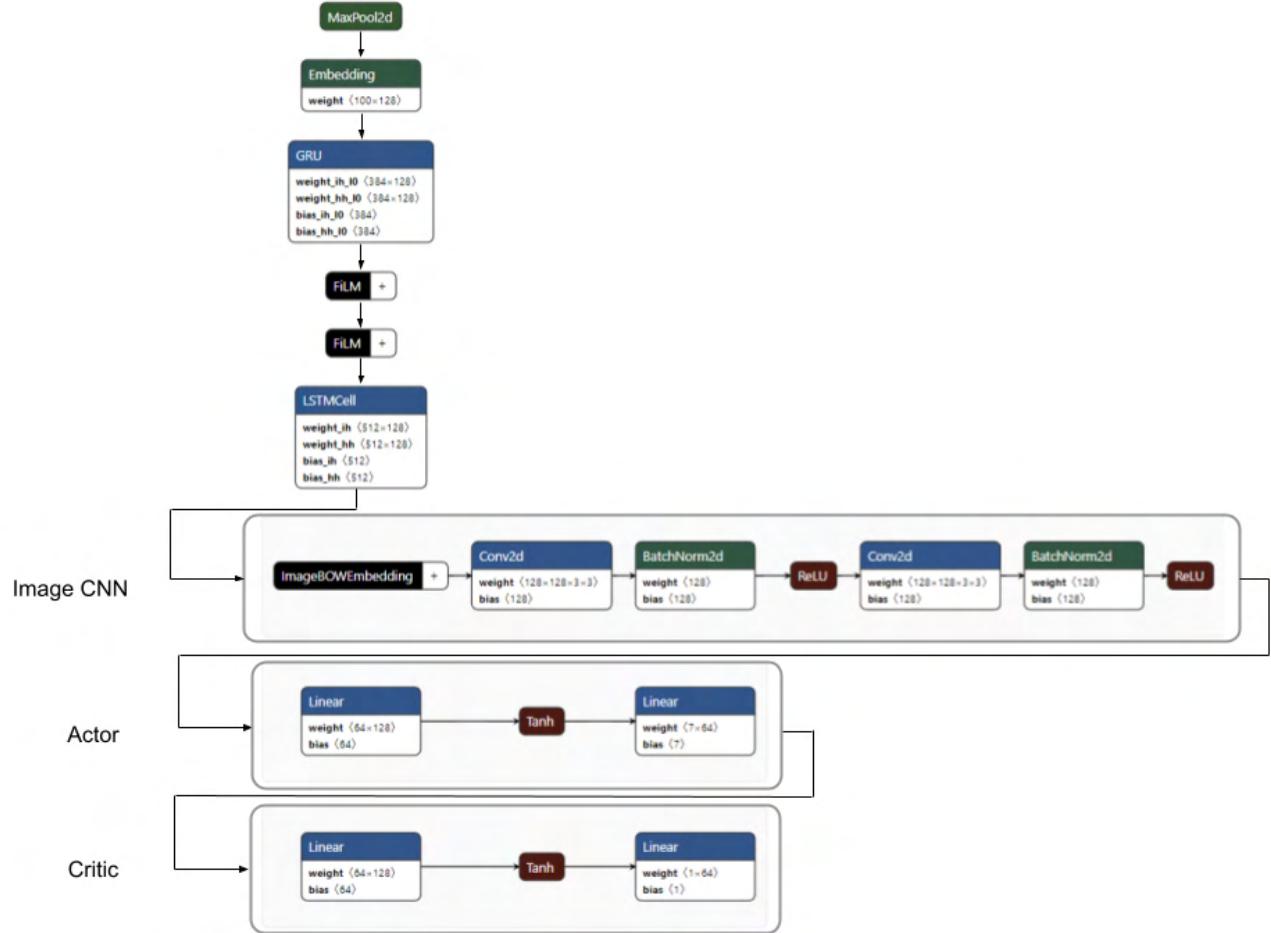


Fig. 6: BabyAI imitation learning model (baseline)

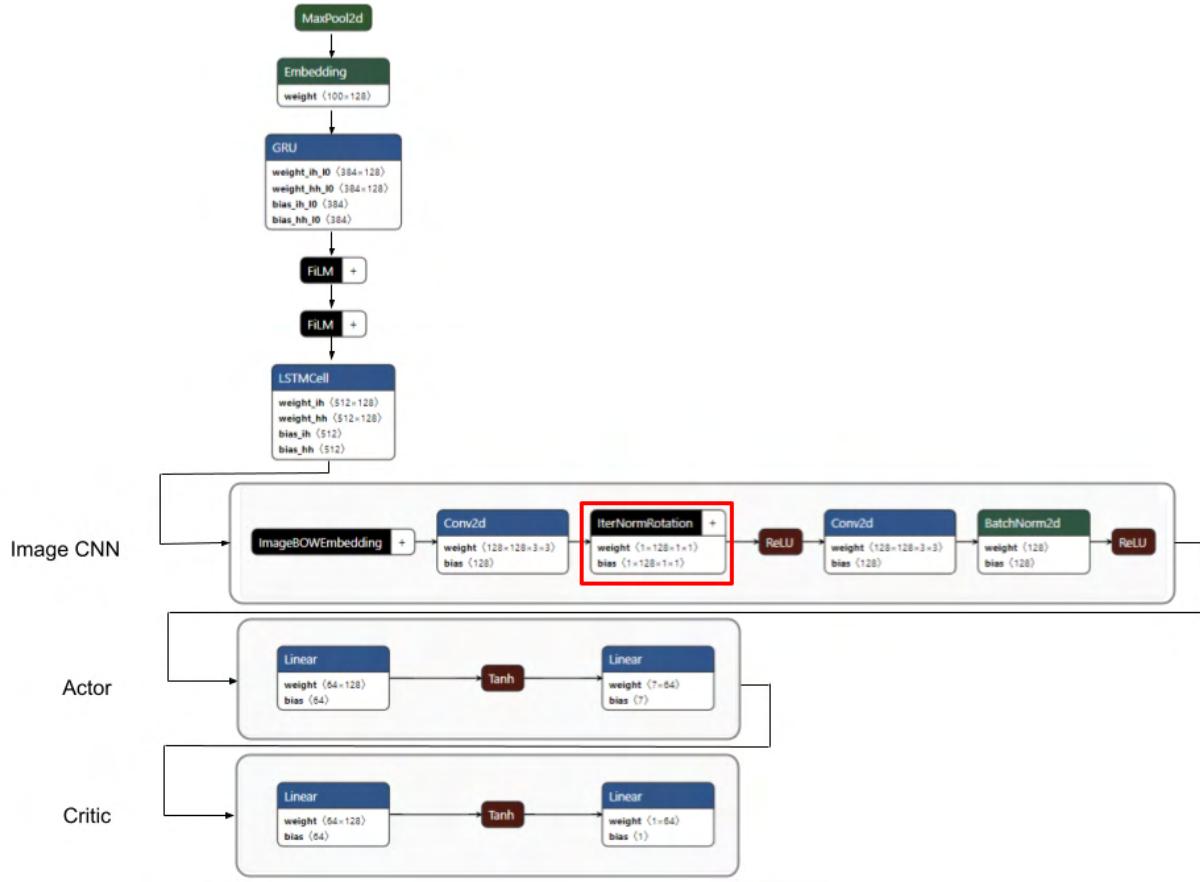


Fig. 7: BabyAI imitation learning model with first batch normalization layer replaced with a concept whitening layer (labeled as IterNormRotation and boxed in red)

Multi-Agent Path Finding for Convex Polygon-Shaped Agents

Nayana Suvarna¹ and Zhongqiang Ren² and Howie Choset²

Abstract— Multi-agent path finding (MAPF) aims to compute collision-free paths for multiple agents between their respective start and goal positions. With most MAPF approaches, the shapes of agents are often inflated to circles or over-simplified as points to reduce the computational complexity of collision checking operations. As a result, valid configurations and trajectories for these non-circular agents may be rejected. To make matters worse, employing an exact collision checking method could greatly slow down MAPF planners as this expensive collision checking method is repeatedly invoked during planning operations. In this work, we explore a collision checking method for convex polygon-shaped agents. We test our method by planning for teams of convex polygon-shaped agents in various grid environments. Through these preliminary tests, we observe that the time for collision checking with our approach is lower when compared to a cell-based collision checking method.

Index Terms— Multi-Robot Systems and Motion and Path Planning

I. INTRODUCTION

Multi-agent path finding (MAPF) aims to compute collision-free paths for multiple agents between their respective start and goal positions. In this work, we strive to plan for teams of agents who carry parts larger than themselves such as poles, beams, and doors while also navigating in cluttered environments such as construction sites. In these scenarios, the shape and orientation of these agents need to be considered to ensure collision-free plans. For applications in simulation and real-world systems, collision checking methods that use the exact shape of an agent may slow down planning. As a result, most MAPF approaches typically oversimplify the shape of these agents as bounding circles or points to defer expensive collision checking operations. But, with these naive methods, valid trajectories may be rejected even if they exist as seen in Figure 1.

In this work, we explore a collision checking approach for convex polygon-shaped agents that can rotate in place. With our approach, we use this convex polygon representation of an agent to determine if the shape of any pair of agents overlaps. During movement, the positions of agents are interpolated for collision checking. To baseline our approach, we compare it to a cell-based collision checking method, where the shapes of agents are approximated by cells on an occupancy grid. For this approach, to determine if a collision exists, we check to see if the approximate shapes of two agents occupy the same cells at a given time. We then test these two approaches using different sets of convex polygon-shaped agents in various grid environments. From our results,

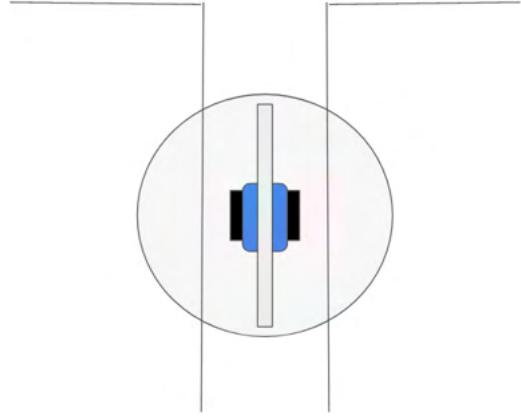


Fig. 1: In this figure, an agent is carrying a large beam through a hallway. Using the traditional method of a bounding circle, a planner would fail to find a solution through the hallway even though one exists

we observe that the time for collision checking with our approach is lower than the cell-based approach.

The rest of this paper is organized as follows: Section II summarizes various related works. After formulating our problem in Section III, we present our approach in Section IV. We show our experimental results in Section V and discuss future works in Section VI.

II. RELATED WORKS

A. Multi-Agent Path Finding (MAPF)

Multi-agent path finding approaches lie on a spectrum from centralized to decentralized approaches. Centralized approaches, such as A*, plan in the joint configuration for all agents. As a result, they offer complete and optimal plans at the cost of poor scalability due to exponential scaling as the number of agents increases. On the other hand, decentralized approaches decompose the search problem into a sequence of single agent problems to provide suboptimal plans with better scalability.

Several approaches have been developed that fall in the middle of the spectrum. Rule-based approaches [1] [2] rely on rule primitives to resolve conflicts between agents. Solutions can be found quickly with these approaches but the plans are not guaranteed to be optimal. Conflict-based Search (CBS) [3] is a two-level search algorithm. The higher level resolves conflicts and adds constraints on the lower level. The lower level then uses these constraints when computing plans for agents.

¹Nayana Suvarna is with the Department of Computer Engineering, University of Pittsburgh, Pittsburgh, PA, USA

²Zhongqiang Ren and Howie Choset are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA USA

Subdimensional expansion [4] is a framework that dynamically increases the dimensions of a search space to resolve conflicts between agents. M* [5] offers complete and optimal plans through combining A* and subdimensional expansion. M* initially computes a set of individual policies for each agent. All agents then follow their individual policies while ignoring collisions with other agents. If a collision is detected, the subset of agents involved in the collision are coupled together and collisions are resolved in this new joint search space. Collision checking serves as an important building block for MAPF problems. For this work, we use our approach to detect collisions between agents which are then resolved by M*.

B. MAPF Collision Checking

MAPF problems broadly consider two main collisions: vertex and edge collisions. Vertex collisions occur when two agents occupy the same node at the same time. Edge collisions occur when two agents approach from opposite directions along the same edge. With simplified descriptions of agents such as circles or points, collisions can be detected by determining if the distance between agents is below a certain threshold.

Many groups, however, have made progress in considering shapes for agents outside of circles or points. Multi-Constraint CBS (MC-CBS) [6], is a variation of CBS that can plan for "large agents" that occupy multiple points at once. During single state expansion, there are more ways for agents to collide with one another compared to when we consider non-point agents. As a result, more intermediate states are expanded and planning time is increased. To circumvent this larger state expansion, MC-CBS introduces multiple constraints to the higher level of CBS to reduce the number of collisions between agents that are within close proximity. Another planning algorithm that uses non-simplified representations for agents is Continuous-Time Conflict Based Search (CCBS) [7]. CCBS combines Safe-Interval Path Planning (SIPP) [8] and CBS to plan for geometrically shaped agents in continuous time.

With these previous works, the authors only consider limited actions for agents which include either translating or waiting in place. With our work, we consider the additional action of rotation in place for agents. With the addition of rotation, we can compute more realistic plans for agents. However, this introduces the additional challenge of considering the swept volume occupied by an agent during rotation for collision checking.

C. Collision Detection Between Shapes

Significant progress has been made in the game design field to develop approaches for fast collision detection between complex shapes. Axis-Aligned Bounding Boxes (AABB) [9] is often used for fast collision detection. With this method, each shape is contained within a rectangular bounding box aligned on the x and y-axis. To check for overlap, a simple check is performed along the x and y-axis to determine if an overlap exists on either axis. Although

this method is fast, it can result in false positives due to the shape of agents being inflated as bounding rectangles. As a result, AABB is often used as a primitive test to determine if agents are within range of one another before more expensive collision checking operations are performed.

Another method for checking for an overlap between shapes is the Separating Axis Theorem (SAT) [9]. This theorem states that an overlap does not exist between two convex shapes if we can find an axis that separates the two shapes. With SAT, a normal axis is first computed relative to every axis of each shape. An overlap check is then done using the projection of each agents vertices on each normal axis to check for a collision. If an overlap exists on every normal axis, then the two shapes are said to be overlapping. With SAT, overlap checks can be done on arbitrarily oriented shapes. For our approach, we need to consider interpolated orientations for agents to approximate the space occupied by agents during rotation. Using AABB checks may result in many false positives, therefore we use SAT to determine if overlap exists between arbitrarily oriented agents.

III. PROBLEM DEFINITION

Let index set $I = \{1, 2, \dots, N\}$ denote a set of N agents. Let $G = (V, E)$ denote a finite graph that describes the workspace shared by all agents, where vertex set V represents a set of possible configurations for agents in $SE(2)$, i.e. $V \subset SE(2)$, and the edge set $E = V \times V$ denotes the set of all the possible actions that can move an agent i between any two adjacent vertices in V . An edge between two vertices $u, v \in V$ is denoted as $(u, v) \in E$ and the cost of an edge $e \in E$ is a strictly positive real value $\text{cost}(e) > 0$.

In this work, we use a superscript $i \in I$ over a variable to represent the specific agent to which the variable belongs (e.g. $v^i \in V$ means a vertex with respect to agent i). Let $\pi^i(v_1^i, v_\ell^i)$ be a path that connects vertices v_1^i and v_ℓ^i via a sequence of vertices $(v_1^i, v_2^i, \dots, v_\ell^i)$ in the graph G . Let $g^i(\pi^i(v_1^i, v_\ell^i))$ denote the cost value associated with the path, which is the sum of the cost values of all the edges present in the path, i.e., $g^i(\pi^i(v_1^i, v_\ell^i)) = \sum_{j=1,2,\dots,\ell-1} \text{cost}(v_j^i, v_{j+1}^i)$. Without loss of generality, to simplify the notations, we also refer to a path $\pi^i(v_o^i, v_f^i)$ for agent i between its initial and final locations as simply π^i . Let $\pi = (\pi^1, \pi^2, \dots, \pi^N)$ represent a joint path for all the agents. The cost value of this joint path is defined as the sum of the individual path costs over all the agents, i.e., $g(\pi) = \sum_i g^i(\pi^i)$.

All agents share a global clock. Each action, either wait or move, for any agent requires one unit of time. When agent i go through an edge $(u, v) \in E$ between time $[t, t + 1]$, let $q^i = q^i(u, v, t, \tau), \tau \in [t, t + 1]$ denote the *motion function*, which describes the configuration of agent i during the transition through edge (u, v) at any time $\tau \in [t, t + 1]$. Note that this definition allows for different motion functions for different agents.

Let $R(q^i)$ denote the *shape function* of agent i which maps a configuration of agent i to a subset of $SE(2)$ that is occupied by agent i , i.e. $R(q^i) \subset SE(2)$. Note that, this

definition allows agents to have different shapes and even varying shapes at different configurations.

For any pairs of agents i, j that move through edges (v_1^i, v_2^i) and (v_1^j, v_2^j) respectively between time $[t, t + 1]$, they are claimed to be in conflict if there exists some $\tau \in [t, t + 1]$ such that $R^i(q^\tau) \cap R^j(q^\tau) \neq \emptyset$.

The objective of the Multi-agent Path Finding in $SE(2)$ is to find a conflict-free joint path π connecting v_o^i, v_f^i for all agents $i \in I$ with $g(\pi)$ at its minimum.

IV. OUR APPROACH

A. Overview

We represent the possible (x, y, θ) positions for agents as vertices in our graph. x and y represent the center coordinates and θ represents the orientation of an agent. Using this reference coordinate, we determine the vertices for the shape of an agent. For this work, we assume that the agents move on a 4-connected grid and can rotate in place.

For a given pair of agents, we use a collision checking function to check for collisions between agents. For our work, we consider collisions to be an overlap between the shapes of two agents. Let v_i and v_j represent two different vertices on our graph G . We define a vertex collision as when agent i_1 is at vertex v_i and agent i_2 is at vertex v_j such that the shapes of these two agents overlap with one another. For edge collisions, we check for an overlap between the shapes of a pair of agents before, during, and after movement. To account for the space occupied during movement, we use an interpolation function that rotates the vertices for the shape of an agent around the (x, y) reference point. Through this function, we can estimate the space occupied by an agent at an arbitrary orientation.

B. Collision Checking

For our approach, we first use a bounding circle check to determine if agents are within range of one another. For vertex collisions, we then use the Separating Axis Theorem [9] to determine if an overlap exists between two agents at the given time. For edge collisions, we determine if an overlap exists before, during, and after movement for agents. We use an interpolation function to determine the approximate shape of an agent during movement.

C. Implementation

For our implementation, we considered agents that occupy multiple adjacent cells along the x or y -axis. An example of these agents planning in a grid environment can be seen in Figure 2. With our implementation, we consider agents of the same size and shape. Additionally, we chose half timesteps for the interpolation in positions of agents.

D. Baseline Approach

To baseline our approach, we compare it to a cell-based collision checking approach that uses the reference point of an agent to determine the approximate cells occupied by the shape of an agent. To determine if a collision exists, a cell comparison is done to see if the shapes of two agents occupy

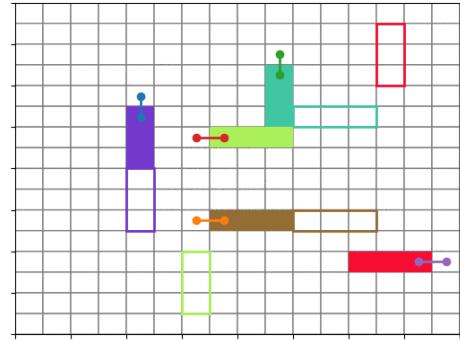


Fig. 2: Visualization of multiple rectangular agents planning in a grid environment. The solid rectangles represent the current positions for agents and the outlined boxes represent the goal positions for each agent. The line pointing outward from each agent represents the direction of forward movement.

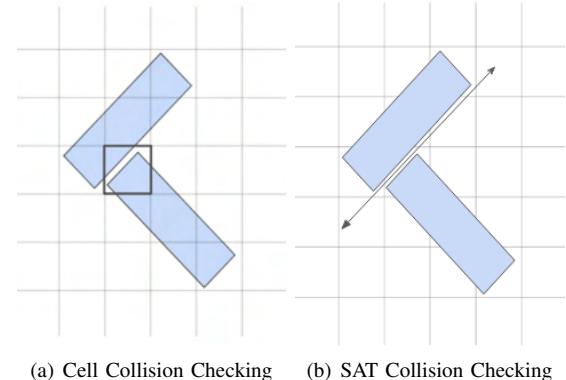


Fig. 3: The cell-based collision checking method on the left classifies the two agents as colliding since the shape of both agents occupy the bolded cell. With our approach on the right, these two agents would not be considered as colliding as an axis exists that separates the shapes of both agents.

the same cell at a given time. Because we're considering rotating agents, this approach also has to consider cells that agents partially occupy. In these cases, we locally increase the discretization of a minimum-maximum bounding box containing the two agents by 4x. To check for overlap we then iterate through these locally discretized cells and use SAT to determine if the shapes of two agents overlap with the same discretized cell.

Because we approximate the shape of an agent with this baseline approach, collisions are classified differently when compared to our approach. These differences can be observed in Figure 3.

V. RESULTS

We implemented our approach and the cell-based collision checking approach using M* as the MAPF planner. We tested sets of 2, 3, 5 and 7 agents with lengths 3 and 5. We ran 50 tests per set of agents and had a 120 second time limit for planning. We tested both approaches using the benchmark 16x16 empty grid world and 64x64 grid world

Map	Method	Success Rate / Path Length / Total Runtime / Collision Checking Runtime			
		K = 2	K = 3	K = 5	K = 7
16x16 empty grid world	Cell for 3-cell agents	1.00 / 12.42 / 1.26 / 1.04	0.88 / 13.90 / 18.09 / 15.50	0.46 / 14.826 / 66.68 / 57.33	0.08 / 17.25 / 111.67 / 95.84
	SAT for 3-cell agents	1.00 / 12.42 / 0.30 / 0.09	0.86 / 13.76 / 17.65 / 6.27	0.30 / 14.6 / 84.17 / 31.16	0.02 / 6.00 / 117.6 / 44.01
	Cell for 5 cell agents	0.94 / 12.31 / 13.73 / 9.43	0.40 / 12.3 / 77.89 / 57.56	0.00 / 0.00 / 120.0 / 57.56	-
	SAT for 5 cell Agents	0.98 / 12.59 / 5.67 / 0.55	0.50 / 12.96 / 65.25 / 9.76	0.00 / 0.00 / 120.0 / 15.07	-
64x64 grid world 10% obstacle density	Cell for 3-cell agents	0.94 / 51.65 / 9.53 / 6.34	0.82 / 58.56 / 22.28 / 16.37	0.54 / 64.88 / 55.89 / 43.94	0.28 / 74.0 / 86.97 / 72.20
	SAT for 3-cell agents	0.94 / 51.65 / 8.70 / 1.25	0.82 / 58.56 / 22.23 / 5.95	0.48 / 65.0 / 63.00 / 17.05	22.0 / 73.27 / 94.07 / 26.52
	Cell for 5 cell agents	0.82 / 64.39 / 21.12 / 16.22	0.54 / 74.51 / 60.02 / 42.85	0.16 / 78.25 / 102.97 / 69.59	0.02 / 61.0 / 119.55 / 84.33
	SAT for 5 cell Agents	0.86 / 63.90 / 20.30 / 1.82	0.52 / 74.03 / 59.44 / 6.54	0.18 / 74.77 / 99.83 / 10.85	0.0 / 0.0 / 120.0 / 14.26

TABLE I: Experimental Results for Cell-Based and SAT-Based Collision Checking

with 10% random obstacle density. From this benchmark set, we randomly generated orientations for agents and only considered sets of valid configurations for agents. We define valid configurations as when the shape of the agent is within bounds of the map.

All our approaches were implemented in Python and ran on a laptop with an Intel® Core™ i7-8550U CPU with 16 GB of RAM. For each set of tests, we measured the success rate (out of 50 tests), average path length, average total runtime, and average runtime for collision checking. The results from our experiments can be found in Table 1.

For the 16x16 grid world, our SAT-based method outperformed the cell-based collision checking method. This can be observed through the higher collision checking runtime, higher overall runtime, and lower success rate for the cell-collision checking method across all tests. Even in the case of five agents where both approaches have a 0% success rate, the runtime for collision checking for our approach is lower than the cell-based one. With the 64x64 grid world with 10% obstacle density, the performance was more similar. For the 3-cell agents, the success rate and path length for sets of three and five agents were the same. But, the overall runtime and runtime for collision checking with our approach was lower. When scaling sets of three and five agents, the cell-based collision checking method had a higher success rate and a lower overall runtime compared to our approach. But, the time for collision checking was significantly larger than our approach. This may be attributed to the differences in the classification of collisions as demonstrated in Figure 3. For the same map with 5-cell agents, overall performance was similar with the runtime for collision checking with our approach being lower than the cell-based method. Because the time for collision checking with our approach was lower than the cell-based method in every test, these results suggest that our approach may scale well when applied to both larger shaped agents and larger sets of agents.

VI. CONCLUSION AND FUTURE WORK

We proposed a collision checking method for multi-agent path finding problems in which the shape of agents can be represented as convex-polygon shapes. We tested our approach by comparing our approach to a cell-based collision checking method and presented some preliminary results. For future work, we plan on testing our approach in more environments as well as with heterogeneous-shaped agents. Another route we could pursue is modifying the search procedure to reduce the runtime complexity of planning.

ACKNOWLEDGMENT

This paper was made with the support of the National Science Foundation under REU Grant 1659774. The authors would also like to thank Rachel Burcin and Dr. John Dolan for their support and hard work in making RISS possible.

REFERENCES

- [1] R. J. Luna and K. E. Bekris, “Push and swap: Fast cooperative pathfinding with completeness guarantees,” in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.
- [2] B. De Wilde, A. W. Ter Mors, and C. Witteveen, “Push and rotate: a complete multi-agent pathfinding algorithm,” *Journal of Artificial Intelligence Research*, vol. 51, pp. 443–492, 2014.
- [3] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, “Conflict-based search for optimal multi-agent pathfinding,” *Artificial Intelligence*, vol. 219, pp. 40–66, 2015.
- [4] G. Wagner and H. Choset, “Subdimensional expansion for multirobot path planning,” *Artificial Intelligence*, vol. 219, pp. 1–24, 2015.
- [5] ———, “M*: A complete multirobot path planning algorithm with performance bounds,” in *2011 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2011, pp. 3260–3267.
- [6] J. Li, P. Surynek, A. Felner, H. Ma, T. S. Kumar, and S. Koenig, “Multi-agent path finding for large agents,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 7627–7634.
- [7] A. Andreychuk, K. Yakovlev, D. Atzmon, and R. Stern, “Multi-agent pathfinding with continuous time,” *arXiv preprint arXiv:1901.05506*, 2019.
- [8] M. Phillips and M. Likhachev, “Sipp: Safe interval path planning for dynamic environments,” in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 5628–5635.
- [9] C. Ericson, *Real-time collision detection*. Crc Press, 2004.

Optimal Control for High-Relative-Degree Systems Under Uncertainty using Exponential Control Barrier Functions

Spencer Van Koevering¹, Yiwei Lyu², Wenhao Luo², John Dolan²

Abstract—Control barrier functions are a formal method that ensures motion safety for constraining autonomous vehicle controllers. However, for general Control Lyapunov Function-Control Barrier Function-Quadratic Programming (CLF-CBF-QP) controllers, the solution feasibility of these driving tasks cannot be guaranteed due to possibly conflicting constraints. In this paper we adapt a method for achieving formally provable safety in high-relative-degree systems using exponential control barrier functions while taking uncertainty into consideration. More importantly, we provide a solution feasibility guarantee in run time using a proposed bi-level optimization framework. The usage of exponential Control Barrier Functions with a high-relative-degree system allows for more robust and general constraints than existing physics-based approaches. Furthermore, we apply this technique to lane changing within the context of autonomous driving and provide experiment results to demonstrate the effectiveness of the proposed technique.

Index Terms—Collision Avoidance, Optimization and Optimal Control

I. INTRODUCTION

Autonomous vehicles will be sharing space with human-controlled vehicles. Safety is critical for any control solution for an autonomous vehicle for this reason. Human drivers can be difficult to predict and hence robust safety measures are needed. Traditional Automated Cruise Control methods are known to lead to aggressive and even dangerous behavior in the pursuit of optimal control [1]. Lane changing is a common scenario that an autonomous vehicle may find itself in that requires tracking multiple other vehicles and maintaining safe distances while performing a turning maneuver.

Control Barrier Functions (CBF) offer a robust method for constraining control of autonomous vehicles and are seeing increased usage [2]. Control Barrier Functions offer a method by which the continuing safety of a system can be guaranteed based upon the forward invariance of a set of safe states [3], [4]. However, if a solution that adheres to the constraints of a CBF

¹Spencer Van Koevering is with the Departments of Computer Science and Mathematics, Whitman College, Walla Walla WA, USA. vankoesd@whitman.edu

²Yiwei Lyu, Wenhao Luo and John Dolan are with the Robotics Institute, Carnegie Mellon University, Pittsburgh PA, USA. {yiweilyu, jdolan}@andrew.cmu.edu, wenhao@cs.cmu.edu

cannot be found, then continuing safety is no longer guaranteed and the search for a safe input fails. Due to this, it is critical to ensure feasibility whenever possible. A method for optimizing barriers proposed by Lyu et al. has been used to ensure that any feasible solution will be found even in stochastic models [2].

While this method is compelling it is limited only to barriers of degree one, in which the state of the vehicle relevant to safety is directly dependent upon the control input [2]. However, desired barriers often have higher relative degree, where multiple derivatives of the state of the vehicle relevant to safety are needed to arrive at an equation depending on input. One situation that can be addressed with a higher-degree extension is that of lane changing under a model where input does not directly affect the position of the vehicle. This sort of model allows for the usage of acceleration, which is a much more accurate representation of the control of a vehicle. First-degree barriers have been used to constrain these sorts of systems [5] using physics based barriers, but a higher-degree barrier would have greater generality and robustness, as explicit physics modelling in the barrier function would no longer be needed. Furthermore, the work done on feasibility guarantees and probabilistic barriers is highly beneficial in a CBF-based controller [2]

In this paper barrier functions of the second-degree will be used to ensure safety of the ego vehicle while lane changing is taking place. Our main contributions are:

- An extension of the first-degree feasibility guarantee from [2] to higher relative degree.
- An extension of the probabilistic framework from [2] to higher relative degree. The second-degree barriers will be guaranteed with exponential Control Barrier Functions.

The methods used in this paper offer a framework with which to widen search spaces for safe control in higher-degree barrier functions.

II. RELATED WORK

CBF validity and applicability to problems like adaptive cruise control lane changing are well known [3]–[5]. An explicit quantitative analysis of solution feasibility conditions and a method for accounting for unbounded

uncertainty are proposed in [2] and [6]. However, the model choice is very simple and does not represent realistic inputs to a vehicle. The kinematic bicycle model is a commonly used model for vehicles, and is robust for how simple it is [5], [7]. This model is consistent at varying speeds if lateral acceleration is kept small [7], [8]. Due to its simplicity, using the kinematic bicycle model allows the construction of more efficient controllers while also remaining robust [7], [8].

The optimization done in [2] is done on a barrier of relative degree 1, meaning that inputs directly affect the state of the vehicle. A position based barrier in the kinematic bicycle model, on the other hand, would be of degree 2 because the second derivative of the position depends on all of the input, but the first derivative does not [3]. In general, for a barrier of degree n the n th derivative of the barrier depends on input and the $n - 1$ th derivative does not. This allows for modelling of acceleration as an input, as opposed to inputting velocity at each time step. The extension of the framework for probabilistic barriers and feasibility guarantees to a degree two system would show the effectiveness of this strategy in more realistic control systems and demonstrate the extensibility of this approach to other high-degree systems.

The kinematic bicycle model has been used to address the lane changing scenario in [5]. However, instead of exponential barrier functions, a more complex physics-based barrier is used [5]. In this barrier, the displacement needed to decelerate and avoid collision is explicitly calculated and included in the barrier, which makes it degree 1. While this does work, an exponential barrier function would provide greater simplicity in the barrier and would be independent of deceleration formulas about the system being used. Furthermore, a feasibility guarantee should also provide the ability for the controller to find more optimal solutions.

III. METHOD

A. Background

A general affine control system takes the form

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})u \quad (1)$$

Given an affine control system of the form 1 where f and g are locally Lipschitz and $\mathbf{x} \in D \subset \mathbb{R}^n$ is the state and $u \in U \subset \mathbb{R}^m$ is the set of admissible inputs [3]. We can define a function $h(\mathbf{x}) : D \subset \mathbb{R}^n \rightarrow \mathbb{R}$, where the set $C = \{\mathbf{x} \in D \subset \mathbb{R}^n : h(\mathbf{x}) \geq 0\}$ is called the safe set [3]. This allows for the barrier function to be defined [3]:

Definition 1. Let $C \subset D \subset \mathbb{R}^n$ be the superlevel set of a continuously differentiable function $h : D \rightarrow \mathbb{R}$, then

h is a control barrier function if there exists an extended class \mathcal{K}_{inf} function α such that for the control system 1

$$\sup_{u \in U} [L_f h(\mathbf{x}) + L_g(h(\mathbf{x}))u] \geq -\alpha(h(\mathbf{x})) \quad (2)$$

for all $x \in D$

This allows for the definition of the set of all inputs that render the set safe as stated in [3] to be

$$K_{\text{cbf}}(\mathbf{x}) = \{u \in U : L_f h(\mathbf{x}) + L_g h(\mathbf{x})u + \alpha(h(\mathbf{x})) \geq 0\} \quad (3)$$

The condition in 2 can also be expressed as $\sup_{u \in U} [h(\mathbf{x}, u) \geq -\alpha(h(\mathbf{x}))]$ [2]. It is proved in [3] that any control that satisfies this condition, will render the safe set invariant.

Since the definition of control barrier function only includes a first derivative, the input of a second-degree affine system, will not be constrained. If we are to use a higher relative degree constraint on the dynamics model (16), then a stronger condition will be needed. The exponential Control Barrier Function (eCBF) as proposed in [3] permits Control Barrier Functions of higher relative degree than one. One issue is that the formal definition of a barrier of degree n , i.e., $h^{(n)}$ depends on u and $h^{(n-1)}$ does not, does not apply cleanly to the desired barrier [3].

Let

$$\eta_b = \begin{bmatrix} h(\mathbf{x}, u) \\ \dot{h}(\mathbf{x}, u) \\ \vdots \\ h^{(n-1)}(\mathbf{x}, u) \end{bmatrix}, F = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}, G = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \quad (4)$$

Where F is $n \times n$, η_b is $n \times 1$ and G is $n \times 1$. Given this, the definition of an eCBF given in [3] is:

Definition 2. Given a set $C \subset D \subset \mathbb{R}^n$ defined as the superlevel set of a n -times continuously differentiable function $h : D \rightarrow \mathbb{R}$, then h is an exponential Control Barrier Function if there exists a row vector $K_\alpha \in \mathbb{R}^n$ such that for the control system 1

$$\sup_{u \in U} [h^{(n)}(\mathbf{x}, u)] \geq -K_\alpha \eta_b(\mathbf{x}) \quad (5)$$

There are some constraints on K_α . If we define:

$$v_0 = h(\mathbf{x}, u)$$

$$v_1 = \dot{v}_0 + p_1 v_0$$

$$v_2 = \dot{v}_1 + p_2 v_1$$

where p are the eigenvalues of $F - GK_\alpha$, then K_α must satisfy

$$\forall p_i = \lambda(F - GK_\alpha) \quad (6)$$

$$p_i > 0 \quad (7)$$

$$p_i \geq -\frac{\dot{v}_{i-1}(\mathbf{x}_0, u_0)}{v_{i-1}(\mathbf{x}_0, u_0)} \quad (8)$$

[3].

B. Probabilistic Barriers and K_α Optimization

The affine controller can be made stochastic by adding random variables as in [2]

$$g_s = [G \quad I], \quad u_s = \begin{bmatrix} u \\ \epsilon \end{bmatrix} \quad (9)$$

The affine controller then becomes:

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g_s(\mathbf{x})u_s \quad (10)$$

Where ϵ is a column vector of n gaussian random variables. Each entry of ϵ will introduce uncertainty to the corresponding entry of the state vector, so many of them may be set with $\mu = \sigma = 0$. The constraints (5) and (8) will now contain random variables. Hence we will have to satisfy it probabilistically

$$P \left(\sup_{u \in U} \left[h^{(n)}(\mathbf{x}, u) \right] \geq -K_\alpha \eta_b(\mathbf{x}) \right) \geq \eta \quad (11)$$

$$P \left(p_i \geq -\frac{\dot{v}_{i-1}(\mathbf{x}_0, u_0)}{v_{i-1}(\mathbf{x}_0, u_0)} \right) \geq \eta \quad (12)$$

Where η is our confidence level. Both of these can be approached with standard probability theoretic techniques.

Remark. When there is one non-zero ϵ , this can be solved up to degree 5 as a regular polynomial. For an arbitrary number of independent ϵ , as long as no term exceeds degree 2 with respect to these random variables, the distribution can be approximated as a Generalized χ^2 distribution [9].

While selection of K_α need only be done at the beginning of the control period, we can choose it at every time step to guarantee the feasibility of the eCBF, while also keeping the entries of K_α as close as possible to some desired values.

$$\min p_i^2 - c_i^2 \quad (13)$$

$$p_i > 0 \quad (14)$$

$$P \left(p_i \geq -\frac{\dot{v}_{i-1}(\mathbf{x}, u)}{v_{i-1}(\mathbf{x}, u)} \right) \geq \eta \quad (15)$$

for $i = \{1, \dots, n\}$, where n is the degree of the barrier. Note that $h^{(a)} : a \geq n$ will depend on u , hence (8) will depend on u which in turn depends on (11). Therefore $\{p_i : i \geq n\}$ will have to be optimized in the same program as u . Depending on the barrier in use, $\{p_i : i < n\}$ may need to be included as well as (8) for all $l \geq i$ may depend on p_i .

Employing both of these techniques will offer a probabilistic safety guarantee and a feasibility guarantee at each time-step.

IV. APPLICATIONS: PROBABILISTIC BARRIERS AND K_α OPTIMIZATION FOR LANE CHANGING

A. Method

Here we take lane changing in a two-lane context using the kinematic bicycle model as an example. The

kinematic bicycle model can essentially be captured with the affine control system

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v \cos(\psi) \\ v \sin(\psi) \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & -v \sin(\psi) & 1 & 0 & 0 & 0 \\ 0 & v \cos(\psi) & 0 & 1 & 0 & 0 \\ 0 & v/l_r & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ \beta \\ \epsilon_1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (16)$$

where x and y are position variables, ψ is the inertial heading and v is speed. a is acceleration in the direction of current velocity and β is side slip, the angle between the current velocity and the longitudinal axis of the vehicle [5]. In order to get the kinematic bicycle model into affine form we assume that β is small and hence that $\cos \beta = 1$ and $\sin \beta = \beta$. While uncertainty in the y direction could be included, the lane changing application depends only on x position, so it is unecessary for the following analysis. The kinematic bicycle affine system is strictly degree 1, as the first derivative does depend on β but not on a . However, in order for acceleration to be considered in a position barrier, a second degree barrier must be used.

In this scenario the ego vehicle must track at most: one vehicle in front of it in the starting lane, one vehicle in front of it in the target lane, and one vehicle behind it in the target lane. Let us denote these f_c , f_t , and b_t , respectively. Preventing collisions with any of these three vehicles can be seen as maintaining some distance in the direction of the road between the vehicles. We can then define our general barrier (assuming the road runs in the x direction)

$$h_m(\mathbf{x}, u) = \left(\left(\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \mathbf{x}_e - \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \mathbf{x}_m \right)^2 - r^2 \right) = (x_e - x_m)^2 - r^2 \quad (17)$$

where e denotes the ego vehicle, m denotes the other vehicle under consideration, and r is some constant that represents a required distance between the centers of mass of the two vehicles.

We also use a trio of Control Lyapunov Functions to implement a nominal controller for the ego vehicle [3]–[5]:

$$V_v(\mathbf{x}) = (v - v_d)^2 \quad V_y(\mathbf{x}) = (y - y_l)^2 \quad V_\psi(\mathbf{x}) = (\psi)^2$$

Where v_d and y_l are the desired velocity and y position, respectively. These functions implement the vehicle's nominal behavior.

Theorem 1. If h_m is an exponential Control Barrier Function for the affine system 16, then the admissible control space $B(\mathbf{x}, u)$

$$B(\mathbf{x}, u) = \left\{ u_e \in U_e : \frac{b_1 - \bar{\Delta}\epsilon_1}{\sigma} \leq \Phi^{-1}(1 - \eta) \right\} \cup \left\{ u_e \in U_e : \frac{b_2 - \bar{\Delta}\epsilon_1}{\sigma} \geq \Phi^{-1}(\eta) \right\} \quad (18)$$

where

$$b_i = -(\Delta_{\dot{x}}) + \frac{\alpha_2(\Delta_x)}{2} + (-1)^{i-1} \sqrt{\frac{-8\alpha_2(\Delta_x)(\Delta_{\dot{x}}) + \alpha_2^2\Delta_x^2 - 2\alpha_1((\Delta_x)^2 - r^2) - 4(\Delta_x)(\Delta_{\dot{x}})}{2}} \quad (19)$$

TABLE I
PARAMETERS FOR OPTIMIZATION OBJECTIVE

c_{ae}	10	$c_{\alpha ft_2}$.0001	c_{δ_3}	1×10^6
c_{β_e}	1000	$c_{\alpha bt_1}$.0001	$c_{\alpha fc_1}$.0001
c_{δ_1}	.001	$c_{\alpha bt_2}$.0001	$c_{\alpha ft_2}$.0001
c_{δ_2}	10000	$c_{\alpha ft_1}$.0001		

TABLE II
VARIABLES FOR OPTIMIZATION CONSTRAINTS

l_{ef}	1	l_{bt_f}	1	g	9.81
l_{er}	1	l_{btr}	1	r_f	$l_{ef} + l_{mr} + 1$
l_{fc_f}	1	limslip rate	$\frac{15^\circ}{t}$	r_b	$l_{er} + l_{mf} + 1$
l_{fc_r}	1	lim _{acc}	.3g	k_v	10
l_{ft_f}	1	lim _{β}	15°	k_y	1
l_{ft_r}	1	μ	.9	k_ψ	1

and $i \in \{1, 2\}$, $\Delta_x = x_e - x_m$, $\Delta_{\dot{x}} = \dot{x}_e - \dot{x}_m$, $\Delta_{\ddot{x}} = \ddot{x}_e - \ddot{x}_m$, with respect to the deterministic affine system (1), $\Delta_{\epsilon_1} = \epsilon_{e1} - \epsilon_{m1}$, $\bar{\Delta}_{\epsilon_1}$ and σ are the mean and standard deviation of Δ_{ϵ_1} , and $K_\alpha = \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}$, will ensure the continued safety of h_m with confidence η .

Theorem 2. If h_m is an exponential Control Barrier Function for the affine system 16, then the admissible space for K_α is

$$p_1 > 0, p_2 > 0 \quad (20)$$

$$p_i = \frac{\alpha_2}{2} \pm \frac{(\alpha_2^2 - 4\alpha_1)^{1/2}}{2} \quad (21)$$

$$\begin{cases} -p_1((\Delta_x)^2 - r^2) - 2\Delta_x\Delta_{\dot{x}} - 2\Delta_x\bar{\Delta}_{\epsilon_1} - \\ \Phi^{-1}(1-\eta)\sigma_{\Delta_{\epsilon_1}} 2\Delta_x \leq 0 & \Delta_x > 0 \\ -p_1((\Delta_x)^2 - r^2) - 2\Delta_x\Delta_{\dot{x}} - 2\Delta_x\bar{\Delta}_{\epsilon_1} - \\ \Phi^{-1}(\eta)\sigma_{\Delta_{\epsilon_1}} 2\Delta_x \leq 0 & \Delta_x < 0 \\ p_1 r^2 \leq 0 & \Delta_x = 0 \end{cases} \quad (22)$$

$$\left\{ \frac{d_1 - \bar{\Delta}_{\epsilon_1}}{\sigma} \leq \Phi^{-1}(1-\eta) \right\} \vee \left\{ \frac{d_2 - \bar{\Delta}_{\epsilon_1}}{\sigma} \geq \Phi^{-1} \right\} \quad (23)$$

where

$$d_i = \frac{-(2p_1\Delta_x + 2p_2\Delta_x + 4\Delta_{\dot{x}})}{4} + (-1)^{i-1} \frac{\sqrt{R}}{4} \quad (24)$$

for $i \in \{1, 2\}$, and

$$R = (2p_1\Delta_x + 2p_2\Delta_x + 4\Delta_{\dot{x}})^2 - 8(2\Delta_{\dot{x}}^2 + 2\Delta_x\Delta_{\ddot{x}} + 2p_1\Delta_x\Delta_{\dot{x}} + 2p_2\Delta_x\Delta_{\dot{x}} + p_1p_2h) \quad (25)$$

where equation 20 is the same constraint as equation 7 since p is not stochastic, and solving equation 8 when a random variable is included gives equation 22 when $i = 1$, and equation 23 when $i = 2$.

These constraints are nonlinear, so quadratic programming as used in [2], [5] is no longer an option, and nonlinear programming (NLP) will be required. With both of these probabilistic constraints defined we can now define the CBF-CLF-NLP optimization problem which will be the controller [10].

$$\min_{\{a_e, B_e, \delta, \alpha\}} c_{ae} a_e^2 + c_{\beta_e} \beta_e^2 + c_{\delta_1} \delta_1^2 + c_{\delta_2} \delta_2^2 + c_{\delta_3} \delta_3^2 + \quad (26)$$

$$c_{\alpha m_1} \alpha_{m_1}^2 + c_{\alpha m_2} \alpha_{m_2}^2 \quad (27)$$

$$|a| \leq \text{lim}_{\text{acc}} \quad (28)$$

$$\pm \cos(\psi + \beta_{t-1}) \leq .5\mu g \quad (29)$$

$$\beta \leq \beta_{t-1} + \text{lim}_{\text{slip rate}} dt \quad (30)$$

$$-\beta \leq -\beta_{t-1} + \text{lim}_{\text{slip rate}} dt \quad (31)$$

$$L_f V_v(\mathbf{x}) + L_g V_v(\mathbf{x}) u \leq -k_v V_v(\mathbf{x}) + \delta_1 \quad (32)$$

$$L_f V_y(\mathbf{x}) + L_g V_y(\mathbf{x}) u \leq -k_y V_y(\mathbf{x}) + \delta_2 \quad (33)$$

$$L_f V_\psi(\mathbf{x}) + L_g V_\psi(\mathbf{x}) u \leq -k_\psi V_\psi(\mathbf{x}) + \delta_3 \quad (34)$$

$$\frac{b_{m_1} - \bar{\Delta}_{\epsilon_{m_1}}}{\sigma} \leq \Phi^{-1}(1-\eta) \vee \frac{b_{m_2} - \bar{\Delta}_{\epsilon_{m_1}}}{\sigma} \geq \Phi^{-1}(\eta) \quad (35)$$

$$p_{m_1} > 0, p_{m_2} > 0 \quad (36)$$

$$\begin{cases} -p_{m_1}((\Delta_{m_x})^2 - r^2) - 2\Delta_{m_x}\Delta_{m_{\dot{x}}} - 2\Delta_{m_x}\bar{\Delta}_{\epsilon_{m_1}} - \\ \Phi^{-1}(1-\eta)\sigma_{\Delta_{\epsilon_{m_1}}} 2\Delta_{m_x} \leq 0 & \Delta_{m_x} > 0 \\ -p_{m_1}((\Delta_{m_x})^2 - r^2) - 2\Delta_{m_x}\Delta_{m_{\dot{x}}} - 2\Delta_{m_x}\bar{\Delta}_{\epsilon_{m_1}} - \\ \Phi^{-1}(\eta)\sigma_{\Delta_{\epsilon_{m_1}}} 2\Delta_{m_x} \leq 0 & \Delta_{m_x} < 0 \\ p_{m_1} r^2 \leq 0 & \Delta_{m_x} = 0 \end{cases} \quad (37)$$

$$\left\{ \frac{d_{m_1} - \bar{\Delta}_{\epsilon_{m_1}}}{\sigma_{\Delta_{\epsilon_{m_1}}}} \leq \Phi^{-1}(1-\eta) \right\} \vee \left\{ \frac{d_{m_2} - \bar{\Delta}_{\epsilon_{m_1}}}{\sigma_{\Delta_{\epsilon_{m_1}}}} \geq \Phi^{-1} \right\} \quad (38)$$

for $m = \{fc, ft, bt\}$, depending on which vehicles are currently being tracked. For deciding when to lane change, we can use a similar approach to that of the finite state machine proposed in [5], and simply test whether the ego vehicle is in the safe set for all relevant vehicles. If so, and if the full NLP is satisfiable, then the lane change maneuver can begin and the full NLP is used until the lane change is complete.

The details of the CLF constraints are beyond the scope of this paper; they implement a nominal controller and details can be read in [3]. However, we have added δ as a slack variable to make these goal-oriented constraints since they are non-critical, as in [5]. The first five constraints are actuation constraints meant to enforce the assumptions of the affine system 16 and kinematic bicycle model. The first constraint represents the maximum allowable acceleration, and prevents the system from breaking physical limits [5]. Constraint 2 keeps β small so the earlier assumption used to derive 16 is valid [5]. Constraint 3 is to keep lateral acceleration below $0.5\mu g$, where μ is the friction coefficient, as the kinematic bicycle model has better validity inside this range [5], [8]. β_{t-1} as the lateral acceleration will be dependent upon the slip from the last time step [5]. Similarly constraints 4 and 5 are limits on how quickly slip angle can change [5].

For all $c, c \in \mathbb{R}^+$, and we can weight them to prioritize different variables. This way feasibility is guaranteed and there is some control over behavior beyond merely safety.

The solution of the step-wise optimization problem serves as the input for the affine controller at that time-

step. One key difference from the degree 1 case is that K_α need only be selected at the beginning of the optimization problem, so at each time step the choice is to use the new optimal K_α or to use the K_α from the previous time step [2]. We also relax r if the problem is not feasible, until at most $r = 0$ to avoid the controller driving itself to infeasibility. This could be more robustly approached with the method in [11], though this is beyond the scope of this paper. We can greedily use the new, optimal, K_α , and default to the choice from the last time step if the problem is infeasible. In either of these cases, safety is still guaranteed [3]. This is done for each eCBF used to constrain the controller.

B. Experiment

The situation used for our experiment was lane-changing on a road with two lanes. This makes the **BL** and **BR** states in [5] unnecessary, as this state was intended to account for a possibilities present by a third lane. In order to initiate a lane change, the controller tests if the ego vehicle is in the safe set for all 3 position barriers, and if there is a feasible solution to the corresponding optimization problem. If so, then lane changing is initiated and all barriers remain in place until lane changing is finished. Before the lane change, only the vehicle in front of the ego vehicle is critical. Therefore the controller attempts to enforce all 3 position barriers to align the ego vehicle for the lane change, but if it cannot, then it defaults to only the front vehicle barrier. After the lane change is complete, only the vehicles in front of and behind the ego vehicle in the new lane are taken into consideration for the constraints. These are the **ACC** and the **L/R** state in [5]. Note that there are cases in which no feasible solution exists, there can be no safe input regardless of what controller we use. For example, if the lane change is finished, and the rear vehicle is travelling faster than the front vehicle, eventually there will be no safe input. There is no input that prevents collision with both the rear vehicle and the front vehicle, as we have no control over these vehicles. This is distinct from a controller being unable to find a safe input. We denote the second-degree controller with probabilistic barriers Prob, and one with deterministic constraints Det. Similarly we if K_α is optimized then we denote the controller Optim, and otherwise we denote it Const.

The use of both probabilistic barriers and K_α optimization had strong positive impacts on safety and optimality of the solutions found by the controller. Probabilistic barriers had a positive impact on collision rate and a small negative impact on feasibility. This was expected, as a probabilistic barrier forces the controller to pick an input deeper into the feasible region to account for motion uncertainty. Overall, safety appears

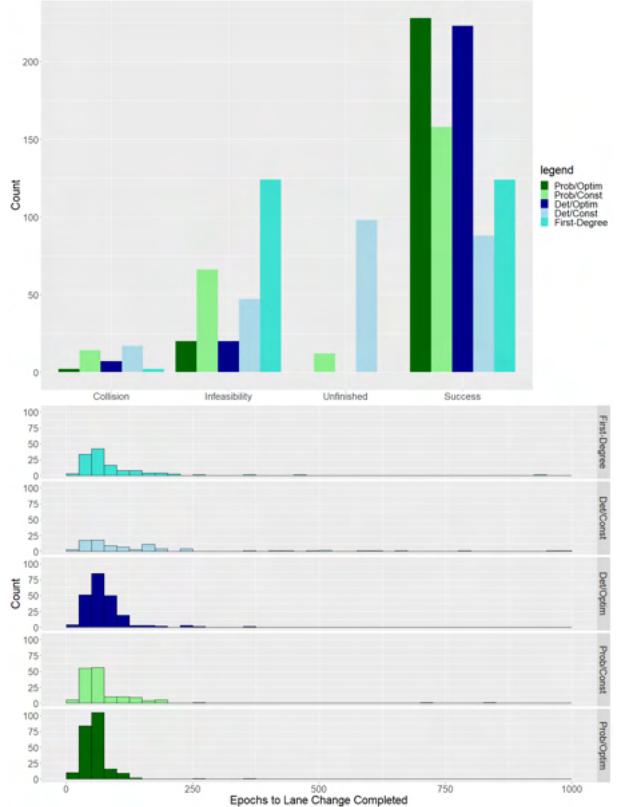


Fig. 1. First-Degree is the controller with first-degree barrier functions from [5]. The other controllers represent our second-degree barrier with K_α barrier constraints and K_α constraints probabilistic or deterministic, Prob/Det, and optimization on or off, Optim/Const. All controllers were governed by a simplified version of the deterministic finite automata used in [5]. The affine controller (16) was used, and non-ego vehicles maintained constant velocity and heading. ϵ was normally distributed with mean 0 and standard deviation .25, and $\eta = .99$ for the probabilistic controller. Each controller was given the same set of 100 randomly generated lane changing scenarios where velocities ranged between 0 and 2.2. Each simulation was run for 10 epochs after the ego vehicle was in the target lane and facing straight forward, or to a maximum of 1000 epochs. The histograms for performance only include instances where a lane change was successfully completed.

to have increased, as the collision rate has drastically decreased and the infeasibility rate is roughly the same when K_α optimization is employed. It is important to note that infeasibility is also an unsafe outcome, as it means that no safe input could be found, though it is not as bad as collision. There also seems to have been some increase in performance, i.e., a decrease in the number of epochs required to complete lane change, when using probabilistic barriers, especially in the case without K_α optimization. It appears that K_α optimization has a strong effect on feasibility in addition to performance, furthermore, only Det/Const and Prob/Const were ever unable to finish the lane change. The impact of K_α optimization on collision was

positive. This was surprising, but it may be the case that this is because with an optimal choice of K_α the ego vehicle will drive minimally aggressively to satisfy its actuation, CBF, and CLF constraints, preventing unsafe scenarios.

The new Prob/Optim barriers had greater safety and better performance than the barriers from [5]. There are the same number of collisions with Prob/Const, and less infeasibility than in the controller using the first-degree barriers. It seems that the control has a very strong condition, that it is able to prevent collision at the same rate as the Prob/Optim controller even under uncertainty. However, its condition may be too strong, which causes infeasibility. The probabilistic barrier, however, is able to be as safe with respect to collisions but also greatly decreases infeasibility. It is interesting that the first-degree barriers from [5] outperformed the Det/Const controller with respect to collision and success rate, as Det/Const is the analogous second-degree choice. This illustrates the nature of the first-degree barrier used in [5]. It is great at preventing collision, but often becomes infeasible. Since these the second-degree controllers had a different objective function than the first-degree controller, performance differences cannot be attributed entirely to the new barriers.

While the comparison of the full Prob/Optim controller to the original first-degree controller in figure 1 depends on the level of uncertainty in the system, we saw increased safety and performance. Though the number of collisions was small for both, the difference is infeasibility was large. Hence the safety difference may be larger than the collision rate suggests. Furthermore, the second-degree controller was able to change lanes more quickly, which indicates its ability to find more optimal control solutions at each time step. The increased safety was unsurprising, as we are accounting for uncertainty. In addition K_α optimization increases the size of feasible regions, which is seen in the comparison to probabilistic controller without K_α optimization.

C. Discussion

While the methods used in [2] do extend to the lane changing problem, some of the features are lost. The ability to separate the problem into a bi-level optimization greatly simplifies both problems, and that is not possible in the second-degree case. Furthermore, constraints become significantly more complex. While deterministic eCBFs may need nonlinear constraints, the quadratic nature of the barrier and of the K_α constraints is unfortunate. In this case, it is not obvious that the degree of this polynomial will increase with the degree of the barrier, so this is mostly a matter of inconvenience. Furthermore our approach in which we discretize the updates of the ego vehicle based on solving

an optimization problem can cause infeasibility at the edge of the safe set, as mentioned in [11].

Despite this, the key features of [2] were retained. Feasibility of the CBF-CLF-QP/NLP is guaranteed if a safe input exists, and probabilistic choices of both u and K_α are possible. Furthermore, the probabilistic and dual optimization does not significantly complicate the NLP beyond what a regular usage of an eCBF barrier would do, as they are often nonlinear anyway.

The approach to probabilistic constraints and optimization of K_α has a profound impact on safety, feasibility, and optimality of the solutions that a controller can find. Probabilistic constraints greatly increased safety of the controller, and also seem to have had a small positive impact on performance, which was unexpected. K_α optimization greatly increased feasibility and increased performance. The combination of these two techniques gave us strictly better constraints second-degree. The first-degree physics based controller performed exceptionally well when compared to the analogous second-degree controller. Despite this initial difference, our experiment shows increased safety and feasibility when using the Prob/Optim barrier relative to the first-degree physics-based barrier, with uncertainty in the system.

V. CONCLUSION

The methods proposed by [2] in order to account for uncertainty and guarantee the feasibility of safe solutions do extend to the second-degree in a general way. While this extension can be inconvenient, depending on the barrier function and affine system in use, the key properties are preserved. We tested this extension and were able to create a powerful controller for an existing problem that outperformed known solutions in situations with uncertainty, and has strong theoretical guarantees while also being a generally applicable method. Furthermore the proposed method for extension works for barriers of arbitrary degree.

VI. FUTURE WORK

The primary drawback of this extension is the move to a single level optimization problem, and this means that characterizing the optimal K_α choice is not as straightforward as in [2]. Some theoretical work to characterize this single level optimization problem would be another direction of future research. Application of the discretized feasibility guarantee from [11] would also increase the ability of our controller to find feasible input, by preventing it from being driven to infeasibility, at the edge of the safe set, as a result of the discretization of time.

APPENDIX

Proof. [Theorem 1]

Taking the first derivative of the barrier with respect to the affine system (16) gives:

$$\dot{h}(\hat{x}, u) = 2(x_e - x_m)(\dot{x}_e - \dot{x}_m + \epsilon_{e_1} - \epsilon_{m_1}) \quad (39)$$

$$= 2 \left(\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \hat{x}_e - \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \hat{x}_m \right) \cdot \left(\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} (f_e + g_e u_e) - \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} (f_m + g_m u_m) \right) \quad (40)$$

and the second derivative gives:

$$\ddot{h} = 2(\dot{x}_e - \dot{x}_m + \epsilon_{e_1} - \epsilon_{m_1})^2 + 2(x_e - x_m)(\ddot{x}_e - \ddot{x}_m) \quad (41)$$

where

$$\ddot{x} = \begin{bmatrix} 0 \\ 0 \\ -v \sin \psi - \beta v \cos \psi \\ \cos \psi - \beta \sin \psi \end{bmatrix} (f + gu) \quad (42)$$

$$= \left(-\frac{v^2 \beta}{l_r} \sin \psi - \frac{\beta^2 v^2}{l_r} \cos \psi + a \cos \psi - a \beta \sin \psi \right) \quad (43)$$

Using the condition in 5, the input needs to satisfy

$$\ddot{h}(\mathbf{x}, \mathbf{u}) \geq -K_\alpha \eta_b(\mathbf{x}, u)$$

Which takes the form:

$$2(\dot{x}_e - \dot{x}_m + \epsilon_{e_1} - \epsilon_{m_1})^2 + 2(x_e - x_m)(\ddot{x}_e - \ddot{x}_m) \geq \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} \cdot \begin{bmatrix} (x_e - x_m)^2 - r^2 \\ 2(x_e - x_m)(\dot{x}_e - \dot{x}_m + \epsilon_{e_1} - \epsilon_{m_1}) \end{bmatrix} \quad (44)$$

Let

$$\Delta_x = x_e - x_m = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \hat{x}_e - \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \hat{x}_m \quad (45)$$

$$\Delta_{\dot{x}} = \dot{x}_e - \dot{x}_m = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} (f_e + g_e u_e) - \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} (f_m + g_m u_m) \quad (46)$$

$$\Delta_{\ddot{x}} = \ddot{x}_e - \ddot{x}_m = \begin{bmatrix} 0 \\ 0 \\ -v_e \sin \psi_e - \beta_e v_e \cos \psi_e \\ \cos \psi_e - \beta_e \sin \psi_e \end{bmatrix} (g_e u_e) - \begin{bmatrix} 0 \\ 0 \\ -v_m \sin \psi_m - \beta_m v_m \cos \psi_m \\ \cos \psi_m - \beta_m \sin \psi_m \end{bmatrix} (g_m u_m) \quad (47)$$

with respect to the affine system **without** uncertainty, so these do not include the ϵ terms. This will allow us to simplify notation, and keep ϵ separate. Let us also denote: $\Delta_{\epsilon_1} = \epsilon_{e_1} - \epsilon_{m_1}$ and $\Delta_{\epsilon_2} = \epsilon_{e_2} - \epsilon_{m_2}$. Isolating the random variable gives:

$$\begin{aligned} 2((\Delta_{\dot{x}}) + (\Delta_{\epsilon_1}))^2 + 2(\Delta_x)(\Delta_{\dot{x}}) &\geq -K_\alpha \left[\frac{(\Delta_x)^2 - r^2}{2(\Delta_x)(\Delta_{\dot{x}} + \Delta_{\epsilon_1})} \right] \\ (\Delta_{\epsilon_1})(\Delta_{\epsilon_1}) + 2(\Delta_{\dot{x}}) - \alpha_2(\Delta_x) &\geq \\ -(\Delta_{\dot{x}})^2 - \frac{1}{2}\alpha_1((\Delta_x)^2 - r^2) - \alpha_2(\Delta_x)(\Delta_{\dot{x}}) - (\Delta_x)(\Delta_{\dot{x}}) & \end{aligned}$$

Applying the quadratic formula

$$\begin{aligned} (\Delta_{\epsilon_1}) &\geq -(\Delta_{\dot{x}}) + \frac{\alpha_2(\Delta_x)}{2} \pm \\ &\sqrt{\frac{-8\alpha_2(\Delta_x)(\Delta_{\dot{x}}) + \alpha_2^2 \Delta_x^2 - 2\alpha_1((\Delta_x)^2 - r^2) - 4(\Delta_x)(\Delta_{\dot{x}})}{2}} \end{aligned}$$

Recall that the input vectors u_e and u_m are only in the \dot{x} and \ddot{x} terms (and therefore $\Delta_{\dot{x}}$ and $\Delta_{\ddot{x}}$). We assume that Δ_{ϵ_1} is a normally distributed random variable. Let its mean and standard deviation be $\bar{\Delta}_{\epsilon_1}$ and Σ . Let

$$\begin{aligned} b_1 &= -(\Delta_{\dot{x}}) + \frac{\alpha_2(\Delta_x)}{2} + \\ &\sqrt{\frac{-8\alpha_2(\Delta_x)(\Delta_{\dot{x}}) + \alpha_2^2 \Delta_x^2 - 2\alpha_1((\Delta_x)^2 - r^2) - 4(\Delta_x)(\Delta_{\dot{x}})}{2}} \end{aligned} \quad (48)$$

$$\begin{aligned} b_2 &= -(\Delta_{\dot{x}}) + \frac{\alpha_2(\Delta_x)}{2} - \\ &\sqrt{\frac{-8\alpha_2(\Delta_x)(\Delta_{\dot{x}}) + \alpha_2^2 \Delta_x^2 - 2\alpha_1((\Delta_x)^2 - r^2) - 4(\Delta_x)(\Delta_{\dot{x}})}{2}} \end{aligned} \quad (49)$$

As this is quadratic, we split into the union of two inequalities:

$$\Delta_{\epsilon_1} \geq b_1 \vee \Delta_{\epsilon_1} \leq b_2$$

This should be true with probability η . We can use the inverse cumulative distribution function Φ^{-1} .

In case 1:

$$P(\Delta_{\epsilon_1} \geq b_1) = 1 - \Phi\left(\frac{b - \bar{\Delta}_{\epsilon_1}}{\sigma}\right)$$

$$P(\Delta_{\epsilon_1} \geq b_1) \geq \eta \iff 1 - \Phi\left(\frac{b - \bar{\Delta}_{\epsilon_1}}{\sigma}\right) \geq \eta$$

$$\frac{b_1 - \bar{\Delta}_{\epsilon_1}}{\sigma} \leq \Phi^{-1}(1 - \eta)$$

In case 2:

$$P(\Delta_{\epsilon_1} \leq b_2) = \Phi\left(\frac{b - \bar{\Delta}_{\epsilon_1}}{\sigma}\right)$$

$$P(\Delta_{\epsilon_1} \leq b_2) \geq \eta \iff \Phi\left(\frac{b - \bar{\Delta}_{\epsilon_1}}{\sigma}\right) \geq \eta$$

$$\frac{b_2 - \bar{\Delta}_{\epsilon_1}}{\sigma} \geq \Phi^{-1}(\eta)$$

□

Proof. [Theorem 2] p_i are defined as the two eigenvalues of $F - GK_\alpha$ [3]. Hence solving for p in general gives

$$p_i = \frac{\alpha_2}{2} \pm \frac{(\alpha_2^2 - 4\alpha_1)^{1/2}}{2}$$

The assignment of the plus or minus term to the different i is arbitrary, we account for both possibilities. The first condition $p_i > 0$ is trivial. The second condition (8) must be split into two cases. Using the alternate formulation, $\dot{v}_0 + p_1 v_0 \geq 0$, from [3], the first case takes the form:

$$\begin{cases} \Delta_{\epsilon_1} \geq \frac{-p_1((\Delta_x)^2 - r^2)}{2\Delta_x} - \Delta_{\dot{x}} & \Delta_x > 0 \\ \Delta_{\epsilon_1} \leq \frac{-p_1((\Delta_x)^2 - r^2)}{2\Delta_x} - \Delta_{\dot{x}} & \Delta_x < 0 \\ 0 \geq -p_1((\Delta_x)^2 - r^2) & \Delta_x = 0 \end{cases} \quad (50)$$

Only the first two cases are stochastic. These give us the constraints:

$$\begin{cases} \frac{-p_1((\Delta_x)^2 - r^2)}{2\Delta_x} - \Delta_{\dot{x}} - \bar{\Delta}_{\epsilon_1} - \Phi^{-1}(1 - \eta) \leq 0 & \Delta_x > 0 \\ -\left(\frac{-p_1((\Delta_x)^2 - r^2)}{2\Delta_x} - \Delta_{\dot{x}} - \bar{\Delta}_{\epsilon_1} - \Phi^{-1}(\eta)\right) \leq 0 & \Delta_x < 0 \\ p_1 r^2 \leq 0 & \Delta_x = 0 \end{cases} \quad (51)$$

Note that the third case is unsatisfiable. Let us expand these to a more intuitive form:

$$\begin{cases} \frac{-p_1((\Delta_x)^2 - r^2)}{2\Delta_x} - \Delta_{\dot{x}} - \bar{\Delta}_{\epsilon_1} - \Phi^{-1}(1 - \eta)\sigma_{\Delta_{\epsilon_1}} \leq 0 & \Delta_x > 0 \\ -\left(\frac{-p_1((\Delta_x)^2 - r^2)}{2\Delta_x} - \Delta_{\dot{x}} - \bar{\Delta}_{\epsilon_1} - \Phi^{-1}(\eta)\sigma_{\Delta_{\epsilon_1}}\right) \leq 0 & \Delta_x < 0 \end{cases} \quad (52)$$

$$\begin{cases} -p_1((\Delta_x)^2 - r^2) - 2\Delta_x \Delta_{\dot{x}} - 2\Delta_x \bar{\Delta}_{\epsilon_1} - \Phi^{-1}(1 - \eta)\sigma_{\Delta_{\epsilon_1}} 2\Delta_x \leq 0 & \Delta_x > 0 \\ -p_1((\Delta_x)^2 - r^2) - 2\Delta_x \Delta_{\dot{x}} - 2\Delta_x \bar{\Delta}_{\epsilon_1} - \Phi^{-1}(\eta)\sigma_{\Delta_{\epsilon_1}} 2\Delta_x \leq 0 & \Delta_x < 0 \end{cases} \quad (53)$$

For the second constraint, let us again use the alternative formulation from [3]:

$$v_1 = \dot{h} + p_1 h = 2(\Delta_x(\Delta_{\dot{x}} + \bar{\Delta}_{\epsilon_1})) + p_1 h$$

The constraint then is:

$$(2)\Delta_{\epsilon_1}^2 + (2p_1\Delta_x + 2p_2\Delta_x + 4\Delta_{\dot{x}})\Delta_{\epsilon_1} + (2\Delta_{\dot{x}}^2 + 2\Delta_x\Delta_{\dot{x}} + 2p_1\Delta_x\Delta_{\dot{x}} + 2p_2\Delta_x\Delta_{\dot{x}} + p_1p_2h) \geq 0 \quad (54)$$

let us apply the quadratic formula as in the proof of theorem 1.

$$d_1 = \frac{-(2p_1\Delta_x + 2p_2\Delta_x + 4\Delta_{\dot{x}})}{4} + \frac{\sqrt{R}}{4}$$

$$d_1 = \frac{-(2p_1\Delta_x + 2p_2\Delta_x + 4\Delta_{\dot{x}})}{4} - \frac{\sqrt{R}}{4} \quad (55)$$

Where

$$R = (2p_1\Delta_x + 2p_2\Delta_x + 4\Delta_{\dot{x}})^2 - 8(2\Delta_{\dot{x}}^2 + 2\Delta_x\Delta_{\dot{x}} + 2p_1\Delta_x\Delta_{\dot{x}} + 2p_2\Delta_x\Delta_{\dot{x}} + p_1p_2h) \quad (56)$$

Then the constraints become:

$$\Delta_{\epsilon_1} \geq d_1 \vee \Delta_{\epsilon_1} \leq d_2 \quad (57)$$

The same approach as used in the proof of theorem 1 can be used to arrive at the final constraints. □

ACKNOWLEDGEMENT

The authors would like to thank the organizers of the CMU Robotics Institute Summer Scholars program for making this work possible. This work was sponsored by the NSF Research Experience for Undergraduates Program.

REFERENCES

- [1] J. Moeckli, T. Brown, B. Dow, L. Boyle, C. Shwarz, and H. Xiong, “Evaluation of adaptive cruise control interface requirements on the national advanced driving simulator,” *NHTSA’s Vehicle Safety Research*, vol. DOT HS 812 172, Aug 2015.
- [2] Y. Lyu, W. Luo, and J. M. Dolan, “Probabilistic safety-assured adaptive merging control for autonomous vehicles,” 2021.
- [3] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, “Control barrier functions: Theory and applications,” 2019.
- [4] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs for safety critical systems,” *IEEE Transactions on Automatic Control*, vol. 62, no. 8, p. 3861–3876, Aug 2017. [Online]. Available: <http://dx.doi.org/10.1109/TAC.2016.2638961>
- [5] S. He, J. Zeng, B. Zhang, and K. Sreenath, “Rule-based safety-critical control design using control barrier functions with application to autonomous lane change,” 2021.
- [6] M. J. Khojasteh, V. Dhiman, M. Franceschetti, and N. Atanasov, “Probabilistic safety constraints for learned high relative degree system dynamics,” 2020.
- [7] J. Kong, M. Pfieffer, G. Schildbach, and F. Borrelli, “Kinematic and dynamic vehicle models for autonomous driving control design,” 06 2015, pp. 1094–1099.
- [8] P. Polack, F. Altché, B. D’Andréa-Novel, and A. De La Fortelle, “The Kinematic Bicycle Model: a Consistent Model for Planning Feasible Trajectories for Autonomous Vehicles?” in *IEEE Intelligent Vehicles Symposium (IV)*, Los Angeles, United States, June 2017. [Online]. Available: <https://hal-polytechnique.archives-ouvertes.fr/hal-01520869>
- [9] A. Das and W. S. Geisler, “A method to integrate and classify normal distributions,” 2021.
- [10] A. D. Ames, J. W. Grizzle, and P. Tabuada, “Control barrier function based quadratic programs with application to adaptive cruise control,” in *53rd IEEE Conference on Decision and Control*, 2014, pp. 6271–6278.
- [11] W. Xiao, C. Belta, and C. G. Cassandras, “Sufficient conditions for feasibility of optimal control problems using control barrier functions,” 2020.
- [12] S. Gottschalk, M. Lin, and D. Manocha, “Obbstree: A hierarchical structure for rapid interference detection,” *Computer Graphics*, vol. 30, 10 1997.
- [13] M. J. Khojasteh, V. Dhiman, M. Franceschetti, and N. Atanasov, “Probabilistic safety constraints for learned high relative degree system dynamics,” in *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, ser. Proceedings of Machine Learning Research, A. M. Bayen, A. Jadbabaie, G. Pappas, P. A. Parrilo, B. Recht, C. Tomlin, and M. Zeilinger, Eds., vol. 120. The Cloud: PMLR, 10–11 Jun 2020, pp. 781–792. [Online]. Available: <http://proceedings.mlr.press/v120/khojasteh20a.html>

- [14] H. Ma, J. Chen, S. E. Li, Z. Lin, Y. Guan, Y. Ren, and S. Zheng, “Model-based constrained reinforcement learning using generalized control barrier function,” 2021.
- [15] C. Ho, K. Shih, J. S. Grover, C. Liu, and S. Scherer, ““provably safe” in the wild: Testing control barrier functions on a vision-based quadrotor in an outdoor environment,” July 2020.
- [16] Q. Nguyen and K. Sreenath, “Exponential control barrier functions for enforcing high relative-degree safety-critical constraints,” in *2016 American Control Conference (ACC)*, 2016, pp. 322–328.
- [17] P. Wieland, “Constructive safety using control barrier functions,” vol. 7, 08 2007, pp. 462–467.
- [18] R. B. Davies, “Algorithm as 155: The distribution of a linear combination of χ^2 random variables,” *Applied Statistics*, vol. 29, no. 3, p. 323, 1980.

Context-Sensitive Template-Based Text Generation for Robotic Agents

Peter Angel Vanegas¹, Roshni Kaushik², Reid Simmons²

Abstract—The efficacy of communication in human-robot interaction scenarios is of increasingly greater interest as robotic agents, and thus these interaction scenarios, grow ever more pervasive in our lives. It does not suffice merely that the correct information is conveyed, but instead that it is delivered along with proper affect, tone, and mood. We hone this into the domain of teachable robots: agents capable of learning new information, ideas, and rules from us, which ultimately they may reciprocate to others. Accordingly, it is within our interest that these human-facing robots are both effective and socially appropriate in communicating with us. To this end, we propose a step towards a socially-sensitive synthesis model, capable of dynamic expression using pre-defined base phrases with context sensitivity by appending appropriate “satellite phrases.” These satellite phrases are sentimentally-tagged modifiers that can be added to base phrases in accordance with desired mood parameters. This templating model allows for some novel elasticity in generating informative responses, which when used in conjunction with other communication aspects (such as facial expressions and body language), creates a more holistic and desirable interaction experience.

Index Terms—human computer interaction, text-synthesis, sentiment analysis, word embeddings

I. INTRODUCTION

From familiar smartphone personalities to human-assistive robots, we are increasingly interacting with, and becoming dependent on, autonomous robotic agents intended to speak and function all their own. Naturally this comes with a set of challenges – it is not so simple that we simply program language comprehension and reasoning skills into an agent, as such an ability is strongly gated by what for humans is layers of inherent grammatical intuition codified by a lifetime of language use. To this end, much of what our autonomous robotic agents say and do is established in pre-programmed dialogue sequences dependent on extensively hard-coded rules, which inherently prohibits a level of flexibility and novelty that is present in natural speech. In essence, these agents tend to speak in repetitive, tonally flat and neutral sequences.^[1]

In recent years, various text-processing tools of interest have arisen that can be appropriated to attend to this issue. Sentiment analysis, which can be used to measure the polarity, or level of positivity / negativity present within a text, and word embedding vectors, which measure the

¹Peter Angel Vanegas is a Computer Science student at Florida International University in Miami, FL. pvane003@fiu.edu

²Roshni Kaushik is a graduate student working under the direction of Dr. Reid Simmons at Carnegie Mellon University’s Robotics Institute in Pittsburgh, PA. roshnika@andrew.cmu.edu

²Dr. Reid Simmons is a Research Professor and director of the Reliable Autonomous Systems Lab at Carnegie-Mellon University in Pittsburgh, PA. rsimmons@andrew.cmu.edu

various levels of relation between discrete words, can be adapted to instead filter, then systematically generate novel phrases that can be leveled against various environmental factors – tone, mood, and emotions that a human interacting with the agent is expressing. Naturally, responses that a robotic agent generates are expected to be sensitive to such aforementioned factors. By utilizing the values returned by these text-processing tools, it is possible to move towards creating agents that have an improved level of sensitivity to sentimentally or emotionally charged scenarios.

In this work, we have so far implemented a system of scoring for word selection in template-based contexts that utilize tags. The program developed allows for a user or agent to feed tagged “maps” of phrases. These tagged maps may contain collections of words to collections of complete phrases (which may themselves have tags embedded within them). Tags correspond to the type of information or part of speech - a tag may be labeled “ADJ” or “ADV” for adjectives and adverbs (respectively), “THANKS” or “GREETINGS” for generic and common immutable phrases, or may be labeled as containing purpose-specific information. These lists may themselves have tags embedded within in the case where certain flexibility may be desired in certain parts of speech, particularly adjectives and adverbs, though this may be extended to verbs and nouns.

In contexts in which it is wanted, scoring is used to select what replaces a tag. Be this a word or a phrase (or a combination of them - again, tags can themselves contain tags), the phrases can be passed through a proposed algorithm that provides a numeric score using Sentiment Analysis and Word Embeddings. These scores can be adjusted to desired effect in order to influence selection in the event we favor a certain level of affect or how semantically related the words in a phrase are. Utilizing this system, we have observed and shown that this process can be used to meaningfully choose words with appropriate tone and affect. However, due to the limitations of our resources, this also at the current lacks human nuance and is currently viable in specific, simple and, and limited scenarios.

II. BACKGROUND

The inspiration behind this work is contingent on the idea that certain parameters of language can be used to supplement text generation approaches. Current deep-learning approaches (such as those found in GPT-3) rely heavily on statistical prediction, and as a result has some practical limitations on maintaining precise affect.^[2] There is, however, work to target different components of language to address this issue. In this project, the aim is to take into

account specific aspects of which sentences and language are composed (tone and semantic relation) and use them as parameters or standards with which affect can be fine tuned to create more genuine and engaging responses. This is achieved through the values provided from sentiment analysis and word embeddings. [3][4]

A. Sentiment Analysis

The sentiment analysis here is performed principally by two open-use python libraries – VADER-Sentiment, and TextBlob. Both packages are readily available online and have been trained in different contexts. TextBlob is a general Natural Language Processing (NLP) library built upon NLTK, based on the pattern library.^[5] VADER-Sentiment is purposed specifically for sentiment analysis, having been trained on social media data. Both tools return a numerical, decimal score that measures the polarity of phrases fed into it, indicating whether a word or phrase is generally perceived positively or negatively.^[6] These tools are yet naïve and lack the ability to reason certain contexts, such as sarcasm, and will fail to reflect these situations, but for the purposes of this work, its general scoring is sufficient.

B. Word Vectors

Word embeddings are reliant on word vectors, which are the result of the vectorization of discrete words into a set of values, represented as dimensions, which then can be used to find similarities in usage, context, and semantics. The similarities between a set of given words are represented by the distance between those set of vectors and their values; extremely similar words generally share a short distance between them, where unrelated words share a large distance between them.

In this project, the Stanford provided Global Vectors for Word Representation (or GloVe) set is utilized. These vectors are trained on a general English corpus scraped from Wikipedia and come in a set dimensions ranging from 50 to 300.^[7] As a compromise on time and accuracy, since the intent of this system is to function in real time with humans, the 50d vectors are utilized to facilitate better load time at the cost of accuracy and diversity of vectors.

III. METHODS

A. Input and Phrase Formation

The system utilizes two main input options: manual sentence entry, and template construction. The former input allows for a user to input a sentence that has been manually tagged (to be elaborated upon) with specific markers to replace specific portions of a statement with the appropriate parts of speech or phrases. This functions as a basic utility for pre-established phrases that may request a level variability in single words, such as adjectives, adverbs, and simple nouns.

Templatic input functions more distinctly. This is intended for the formation of longer, more diverse phrases and even set of sentences by drawing from a list of curated “core” phrases that may themselves have embedded tags for changing various smaller parts of speech within a larger phrase.

Examples of these core phrases can include game rules, system notifications, and advisory text. These can themselves be modified by satellite phrases, modifying phrases that serve some effect on changing the delivery of tone, mood, or information in the text, that are meant to fit around the core piece of informative text.

Tags generally take the form of either an open part of speech (nouns, verbs, adjective, and adverbs), or type of phrase which serves a specific function. These functions depend on the intent behind of the dialogue designer, i.e., someone who is curating phrases for the specific purpose of this system. A phrase may serve to deliver a comment on a specific object or action a user takes, functioning discretely on its own. This is again an example of a “core” phrase. Some phrases may serve as modifiers that are unable to stand on their own. Suppose the following examples:

“*Your performance was subpar.*”

“**I think** Your performance was subpar.”

The former phrase is an objective statement of one’s performance, but may be interpreted as a blunt statement and may be inconsiderate or inappropriate of the environment in which the comment was made. The latter phrase is a “softened” by the addition of a personal (that is, using the 1st person singular pronoun ‘I’) modifying phrase snippet. The addition of phrases like these is generally arbitrary but meaningful and diverse and attunes to the attitudes at hand. In an active educational or advisory scenario, it may be beneficial to address students with softer, more personalized tone in the event of high levels of frustration or stress, rather than blunt statements of performance (which of course, may have their own use).

Other, smaller phrases can be substituted and written into the aforementioned place. “I believe,” “I feel [like],” “It seems,” may substitute this the place of “I think,” and these are generally left to the discretion of the dialogue designer.

Conversely, this system can also be used to generate more assertive phrases. As stated before, an unmodified statement as in the previous example may be considered blunt, but not necessarily assertive or imperative. In this scenario, consider:

“*Leave the building.*”

“**Please** leave the building.”

“*Leave the building immediately.*”

“**You must** leave the building now.”

These smaller, non-core satellite phrases can be utilized to create a level of imperativeness or urgency.

Suppose a chess-based game rule: “Protect the king.” This rule may be harbored in a file that reacts to the tag [RULE-KING]. In the event it is detected that the user is frustrated, a program interacting with the phrase mapping system may deem it appropriate to add a preceding softening phrase, which itself is harbored with the tag [SOFT]. It should be noted that the position of these phrases is grammatically significant; meaningful organization of tags is necessary

(for the moment, there is a tentative internal system being developed to indicate necessary positioning that can be manually codified). Appropriately, [CONSEQUENCE] may be required as well to convey the result of recent or further actions. The combination of these are be sent as an array to fit in phrases appropriately, taking the form of

[SOFT] [RULE-KING] [CONSEQUENCE]

- which itself can, once phrases are chosen and fitted, ultimately create a phrase like “Please protect the king, or else you will lose,” or “Remember to protect the king, otherwise the game will end.” The choice and wording of curated satellites currently must be deliberate to facilitate grammaticality and clarity. It should be noted phrase maps do not require grammatically open phrases for generating responses, and can utilize stand-alone phrases as well (for example, greetings like “Hello,” or “Good morning,” or expressions of thanks like “Thank you” and “You’re welcome”).

Utilizing the previous examples, we may see those examples can be generated from phrase maps like the following:

[SOFT] [PERFORMANCE]

[IMPERATIVE] [EVACUATE] [TIME]

Which would generate the “I think your performance was subpar” and “You must leave the building now” examples respectively. Additionally, tags may themselves have embedded sub-tags that perform certain functions as well, such as declaring that a set of generated phrases or specific modifications must be scored to fit the scenario, or declaring that we would like to score the semantic distance between a set of modifying words and a target word. Taking for instance the [PERFORMANCE] example, the phrase may be saved in text as so:

*Your performance was **ADJ**.*

Where ****ADJ**** indicates that an adjective from the adjective set is meant to take its place. Tags inside of other tags (subtags) are currently identified using two surrounding asterisks. In most scenarios, this modifying word is also scored, as it must of course fit the requested sentiment for the context. The process of scoring and selection is described in the next subsection.

B. Scoring and Selection

A standard must be applied in order to filter and correctly choose what words and phrases are most appropriate in a given context. This is done by utilizing sentiment analysis and word vectors. If a tag contains the appropriate signal (currently, a “+” in the tag name), then it generates a set of phrases using the contents of that tag and sends them to be selected according to a mood parameter. A mood parameter at the present is a simple value, arbitrarily decided by integers -2 to 2, with -2 meaning “strongly negative,” 0 “neutral,” and 2 “strongly positive.” -1 and 1 are moderately negative or positive, respectively. TextBlob and VADER-Sentiment both return decimal values independent from the

mood parameter’s integers (which function only as stand-ins to call for the desired tone of sentiment). TextBlob returns decimal values from -1 to 1 (negative to positive sentiment respectively); VADER returns values from 0 - 1, separated by how strongly they lean in either positive, negative, or neutral sentiment, all of which are factored into a *compound* score, of which we utilize rather than the previous 3.

For selection of moderately tonal sentiments utilize a score range from 0.00 - 0.65, while strongly tonal sentiments will favor sentiments at strictly at .5 and greater, however, these boundaries are adjustable in code by a dialogue designer. These values are negated for negative sentiments. Neutral sentiments generally are scored between the interval of -0.4 to 0.4. From a selection of phrases closest to the desired affect, a variable, arbitrary number of them that best fit are chosen, and from those, a random completed phrase is chosen to add a level of novelty to prevent monotony in responses.

At a more thorough level, specific words and parts of speech can also be targeted if desired, although these must be manually tagged within the design of the phrases as well. Translating this to our program, we can opt to modify a noun with an adjective, or an adjective with an adverb.

When determining an appropriate modifier, the sentiment is taken into account along with the semantic distance of the modifying word from the target word. The semantic distance of words is a decimal within the range inclusively of 0.0 to 8.0 or above (results above 8 are possible, but unseen given current conditions), and are normalized to produce d according to the following formula, where delta is the semantic distance:

$$d = 10 - \frac{1}{1 - \log_8(\delta)} \cdot 0.1 \quad (1)$$

This reduces scores to a decimal between 0 and 1. Beyond an input distance of 6.498, these scores (along with their associated word) are discarded for being determined to be significantly irrelevant. This value is not arbitrary; An input of 6.498 for delta is the zero of this expression that tends downward, meaning values at this or above produce results that are negative and thus discarded for their supreme irrelevancy.

Sentiment scores are then averaged, and distance from the mean for a given word’s sentiment score is used to determine the weight or “score of the score.” This value, s , is then summated with the normalized semantic distance, d , weighted by a hyperparameter α that is arbitrarily chosen to decide whether semantic distance or sentiment distance is valued more (this, again, is left to the designer or writer’s specifications).

$$s \cdot \alpha + d \cdot (1 - \alpha) \quad (2)$$

Scores for a set of words are aggregated, placed into a phrase’s appropriate tag, and those set of phrases can potentially be scored as well. An arbitrary n amount of the best fit are chosen (for our experiment, we set $n = 5$), from which a random phrase is selected for output. A smaller n

seeks to choose from the absolute best fit, while a larger n broadens the acceptable range.

IV. RESULTS

The utilization of sentiment analysis and semantic distance shows promise and displays desirable variance in its selection process. Precisely, it can be seen that varying the weight of semantic distance versus sentiment score can produce results that change outcomes. As seen in figure 1, variance in the alpha hyperparameter will vary the scoring and selection of certain words. Some words consistently appear, while others do not appear until the parameters have been sufficiently adjusted. The examples provided here are additionally the sample of words which appeared; various appropriate words were omitted by the system altogether, words such as "splendid" or "incredible."

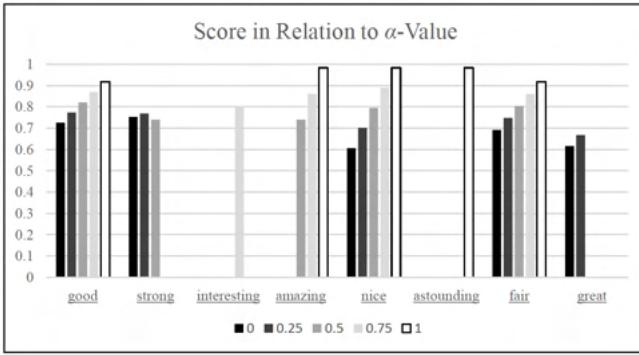


Fig. 1. The word targeted for modification was "move," aiming for a moderately positive sentiment. Some elementary, common words like "good" and "nice" appear frequently regardless of alpha value. As alpha increased, which biases towards sentiment rather than semantic distance, more strongly positively sentimental, rather than close-by-vector distance words were chosen, such as "astounding," or "amazing."

The selections made by this process fit to a generalized idea of sentiment (the concept of being "positive," "negative," or "neutral"), but even provided this scoring system, it sometimes may favor words that do not accurately fit the tonal context. It is possible for it to use superlative words in mild scenarios (for example calling something "amazing" when an action was only satisfactory), and this lack of nuance and may cause awkward or tone-inappropriate responses. Regardless, the level of flexibility in response generation for a set of circumstances is beneficial, and if necessary, a dialogue designer can choose to tighten its selection criteria by changing the aforementioned mood parameters (see Methods: *Scoring and Selection*), but presents another hurdle in user friendliness and is currently detrimental to diversity of phrase generation.

V. CONCLUSIONS AND FUTURE WORK

The system handles construction of phrases using its phrase maps well, however, faces issues of necessary grammaticality. Phrases must be heavily manually curated to assure grammaticality, as no grammar checks are implemented to assure for features such as agreement or orthographic features such as capitalization. This required "grammatical

smoothing" currently presents some challenges to user / designer friendliness and is to be taken for further consideration.

This system is not intended for independent use and is meant to be utilized with separate systems that are capable of creating more holistic experiences. For example, when creating agents that are robotic tutors, one possibly able to detect our emotions and attitudes, which may have systems to determine body or facial language. In this respect, although there are limitations on dialogue creation, it nonetheless potentially adds a layer of novelty to such experiences. In addition, although the processes shown here are trained on general English corpora, these tools can potentially be trained on specific ones (for example, product reviews or educational materials) to hone in on the purpose of the grander projects.

Its main potential benefits lie within the simplification of dialogue creation; although writers may still be required to curate specific phrases and terms to be used, instead of manually hard-coding or typing in a corpus of phrases, this allows for recombination of phrases that are adaptive to specific scenarios by calling upon appropriate phrases. Although this yet is a level away from true language comprehension, the systems described herein are a step towards creating more comprehensive and dynamic robotic interaction environments.

ACKNOWLEDGMENT

The work presented here is supported NSF Grant #1659774 and the Reliable Autonomous Systems Lab (RASL) headed by my mentor Dr. Reid Simmons, and his graduate student Roshni Kaushik. Special thanks are extended to Rachel Burcin & Dr. John Dolan, coordinators of the RISS 2021 program, and Allison Moore, my labmate through this experience.

REFERENCES

- [1] T. Goswamy, I. Singh, A. Barkati, and A. Modi, "Adapting a language model for controlled affective text generation," in *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 2787–2801. [Online]. Available: <https://aclanthology.org/2020.coling-main.251>
- [2] L. Floridi and M. Chiratti, "GPT-3: Its nature, scope, limits, and consequences," *Minds and Machines*, vol. 30, no. 4, pp. 681–694, Nov. 2020. [Online]. Available: <https://doi.org/10.1007/s11023-020-09548-1>
- [3] A. I. Alharbi and M. Lee, "Combining character and word embeddings for affect in arabic informal social media microblogs," in *Natural Language Processing and Information Systems*, E. Métais, F. Meziane, H. Horacek, and P. Cimiano, Eds. Cham: Springer International Publishing, 2020, pp. 213–224.
- [4] Z. Zeng, Y. Yin, Y. Song, and M. Zhang, "Socialized word embeddings," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, 2017, pp. 3915–3921. [Online]. Available: <https://doi.org/10.24963/ijcai.2017/547>
- [5] S. Loria, "Textblob: Simplified text processing," *Release 0.16*, 2020. [Online]. Available: <https://textblob.readthedocs.io/en/dev/>
- [6] C. Hutto and E. Gilbert, "Vader: A parsimonious rule-based model for sentiment analysis of social media text," 01 2015.
- [7] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. [Online]. Available: <https://aclanthology.org/D14-1162>

Toward the Use of Model Predictive Control for Quadrotor Flight

Fausto Vega¹ and Zachary Manchester²

Abstract—In this paper, we present promising results towards the use of optimal control for a quadrotor. One method is model predictive control (MPC) which utilizes the model and dynamics of the quadrotor to select an optimal control action over a set horizon. MPC also allows constraints on the environment and hardware to be added to the control scheme which allows for safe maneuvers at every time step. A quaternion-based approach for rotations will also be implemented to avoid singularities that happen with Euler angles. Many roboticists use the PX4 Pixhawk as their flight controller which uses PID controllers to stabilize the quadrotor and does not allow for aggressive maneuvers. Other optimization methods such as a Linear Quadratic Regulator (LQR) controller were also implemented on a quadrotor model as it is unconstrained MPC. However LQR can lack robustness for aggressive trajectories and experience singularities. Experimental results show that the LQR controller ensures safe trajectories for simple paths over a discrete time. MPC options were also explored, and these results converged to their goal at a faster rate in a safe manner.

Index Terms—Optimization and Optimal Control, Aerial Systems: Mechanics and Control

I. INTRODUCTION

Quadruped robots have rapidly advanced in control and design within the last decade due to their interest in use in several applications. Some of these applications include disaster response [1], automated delivery [2], and agriculture [3]. Their dynamics allow them to be controllable which enables quadruped robots to follow a reference trajectory with precision. Common control systems on a quadrotor include PID control, yet this type of control does not take into account obstacles and safety constraints of the system. Some safety constraints while flying a quadrotor are the torque limits in the motors as well as the boundary from the flying zone. MPC treats the control of the quadrotor as an optimization problem and allows safety constraints such as torque limits to be implemented in the problem. To accurately arrive to a goal state, a quadrotor must perform three dimensional translations and rotations. A three dimensional representation of a rotation that utilizes Euler angles experiences singularities and leads to less degrees of freedom due to a gimbal lock. However, this is avoided by representing rotations as quaternions and Rodriguez parameters which allow rotations to be represented through linear algebra and vector calculus. Other methods for optimal control rely on differential geometry and Lie group theory which are non trivial and not common to roboticists.

¹Fausto Vega is with the University of Nevada, Las Vegas, 4505 S Maryland Pkwy, Las Vegas, NV 89104, USA
faustovega10@gmail.com

²Zachary Manchester is an Assistant Professor at The Robotics Institute, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213, USA
zacm@cmu.edu

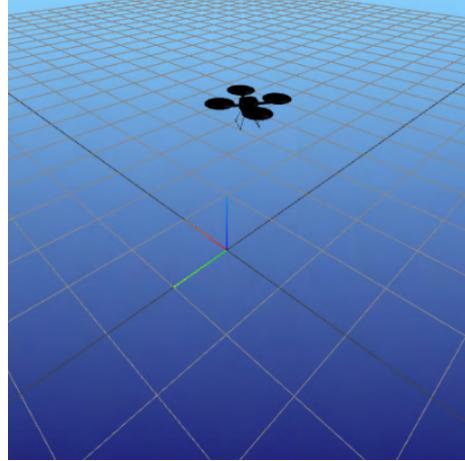


Fig. 1. Quadrotor Hovering in a Julia Simulation Environment

Other control systems have been implemented on quadrotors and have shown positive results. Kumar and Mellinger developed optimal trajectories in real time for the control of quadrotors through corridors [4]. Aggressive maneuvers were also implemented by Kumar and the GRASP Lab by designing optimal feasible trajectories and controllers [5]. Yet, both of these controllers used Euler angles to express the attitude of the quadrotor which may lead to gimbal lock at certain states. Bouabdallah presented a combination of a PID and backstepping technique guaranteed asymptotic stability and robustness while switching states [6]. However, a disadvantage is an external sensor had to be used to avoid colliding into the walls which is not a problem with MPC as the constraints are part of the optimization problem.

Model predictive control with nonlinear dynamics is also a popular subject among control theory. Kim presented the use of a nonlinear MPC controller for position and heading tracking of a helicopter [7]. The optimization method used was gradient descent which can be subject to overshoot and cause a zig-zag behavior when finding the minimum. Real time solvers have been developed to avoid these errors and solve optimization problems extremely fast. State-of-the-art trajectory optimization solvers such as ALTRO have been developed to find a solution quickly [8].

In this paper, we study the use of a LQR controller and a model predictive control (MPC) algorithm to drive a quadrotor to a desired state. This state includes the position, orientation, linear velocity, and angular velocity of the quadrotor. Control using a linear quadratic regulator (LQR) was implemented and results were collected to compare to the MPC solution once results are collected. The communi-

cation between the flight controller and the on board LQR controller was also studied for efficient data transmission. MPC is not common among flight controllers as this method only guarantees convergence when the linearized model is fully controllable. Yet, in this work, we linearize the dynamics model of the quadrotor about a hover position and obtain a controllable system. However, the linearized model utilizes small angle approximations that are only valid at small roll and pitch angles. Therefore, simple trajectories will be executed to ensure the validity of the linearization. This paper also focuses on the modeling of the quadrotor using quaternions to avoid the singularities experienced using Euler angles. The quadrotor position data will be shown for each controller to show the speed and robustness of the controller.

The remainder of the paper is organized as follows: Section II describes the dynamics of the quadrotor, the method of data transfer between multiple components, and the experimental setup. Section III will present the results of the LQR controller and MPC controller, and Section IV will address conclusions and future work on this project.

II. METHODS

A. Dynamics

The quadrotor dynamics will be derived in this section. A quadrotor model is shown in Figure 2 to depict the forces (T_i) and moments (M_i) where $i = 1,2,3,4$. Each motor produces a thrust and a moment which are modeled with Equations 1 and 2. In these equations k_f and k_m represent the thrust and moment coefficients and they obtained using thrust test data provided by the motor manufacturer.

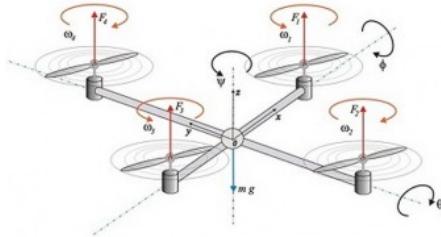


Fig. 2. Quadrotor Forces and Moments

$$T_i = k_f u_i \quad (1)$$

$$M_i = k_m u_i \quad (2)$$

The state of the quadrotor consists of 13 elements which are position in the inertial frame (x, y, z), a quaternion (q_x, q_y, q_z, q_w), linear velocity in the body frame (v_x, v_y, v_z), and the angular velocity in the body frame ($\omega_x, \omega_y, \omega_z$). Equation 3 depicts a summary of the elements in the state vector and Figure 3 shows the frames and notation of the quadrotor.

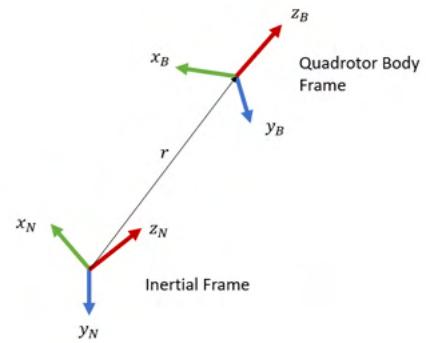


Fig. 3. Quadrotor Coordinate Systems

$$x = \begin{bmatrix} r^N \in \mathbb{R}^3 \\ q \in H \\ v^B \in \mathbb{R}^3 \\ \omega^B \in \mathbb{R}^3 \end{bmatrix} \quad (3)$$

Next, the kinematics of the quadrotor will be discussed as these equations map the velocities to the position. The kinematics related to the position are shown in Equation 4. Since the position is expressed in the body frame, a rotation matrix or quaternion is needed to transform the velocity measurement to the inertial frame (N).

$$\dot{r}^N = Qv^B \quad (4)$$

The derivative of the quaternion is shown in Equation 5. The notations for the L, H, and the skew symmetric operator ($[x]^\times$) are shown in the following equations (5,6,7). In Equation 7, q_s refers to the scalar part of the quaternion ($q_s \in \mathbb{R}$) and q_v represents the vector part ($q_v \in \mathbb{R}^3$). This notation is established in a prior paper published by the Robotics Exploration Laboratory at Carnegie Mellon University [9].

$$\dot{q} = \frac{1}{2}L(q)H\omega^B \quad (5)$$

$$H = \begin{bmatrix} 0 \\ I_3 \end{bmatrix} \quad (6)$$

$$L(q) = \begin{bmatrix} q_s & (-q_v)^T \\ q_v & q_s I + [q_v]^\times \end{bmatrix} \quad (7)$$

$$[x]^\times = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \quad (8)$$

The translational dynamics of the quadrotor follow Newton's second law of motion. However, rotations were applied to obtain the force and acceleration in the body frame, and this results in an additional term to the standard second law. This relation is shown in Equation 9.

$$\dot{v}^B = \frac{1}{m}F^B - \omega^B \times v^B \quad (9)$$

The forces that the quadrotor experiences are expressed in Equation 10. Gravity is the first force and it is rotated to obtain the result in the body frame. The following term corresponds to the thrust force generated by the 4 motors onboard the quadrotor which is expressed in Equation 1.

$$F^B = Q^T \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ k_t & k_t & k_t & k_t \end{bmatrix} u \quad (10)$$

Finally, the rotational dynamics are described by Euler's equation (Equation 11). J corresponds to the inertia matrix which is dependent on the quadrotor size and mass. A 3D model of the quadrotor was designed on Solidworks to obtain the inertia matrix using the mass properties function. The variable τ represents the total torque of the system which comprises of two sources. The first source is the thrust (Equation 1) multiplied by the distance (l) between the motor and the center of gravity, and the second is the propellers as they generate a torque with the moment coefficient (shown in Equation 12).

$$\tau^B = J^B \dot{\omega} + \omega^B \times J \omega^B \quad (11)$$

$$\tau^B = \begin{bmatrix} lk_t(u_2 - u_4) \\ lk_t(u_3 - u_1) \\ k_m(u_1 - u_2 + u_3 - u_4) \end{bmatrix} \quad (12)$$

B. LQR Control

The Linear Quadratic Regulator (LQR) is a common method in control theory that provides feedback gains that ensure a closed loop stable system. This method only applies to linear systems in the following form.

$$\dot{x} = Ax + Bu \quad (13)$$

Using the Runge-Kutta iterative method, the future states in discrete time can be estimated. A Taylor series expansion can also be applied to the state and result in Equation 14. This linearization results in the A and B matrices being the partial derivative of the dynamics with respect to the state and the control. For this experiment, the A and B matrices were linearized about a hover position which accounted for gravity.

$$\delta x_{k+1} = \frac{\partial f_k}{\partial x} \delta x + \frac{\partial f_k}{\partial u} \delta u \quad (14)$$

An attitude Jacobian was then applied to the quaternion part of the state as simply linearizing a system with a quaternion state results in an uncontrollable linear system. This is because the quaternion causes the A matrix to be rank deficient due to the quaternion unit norm constraint. To determine whether the system was controllable, the controllability matrix was used.

The cost function that is minimized throughout a discrete time step is shown in Equation 15. Q and R represent constant matrices that act as gains to tune the system depending on the characteristics needed. The Q matrix must be positive

definite and the R matrix must be positive semi-definite for LQR to work.

$$\min \sum_{k=1}^{N-1} \frac{1}{2}(x_k)^T Q_k x_k + \frac{1}{2}(u_k)^T R_k u_k + \frac{1}{2}(x_N)^T Q_N x_N \quad (15)$$

Figure 4 shows the feedback loop. The error state is multiplied by the K matrix and drives the system to the reference state. The goal of the system is to drive the error state to zero. In this case, the reference state is a hover position and the initial position is a random position and orientation in the air.

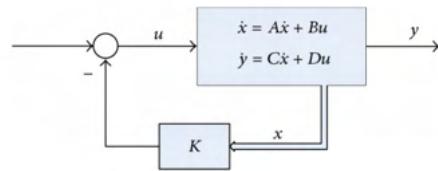


Fig. 4. LQR Control Diagram

C. Model Predictive Control

Model predictive control utilizes the model of the system to predict the future output behavior. Using this prediction, the optimal control problem is solved. Constraints on the inputs and states can be set and the problem will take these relations into account when solving the optimization problem. For example, torque limits are a potential constraint to not have unrealistic PWM commands to the motor. The optimization problem for MPC is shown in Equation 16. H represents the horizon of the problem which is the discrete time the plant has to reach the reference state. The x_N term is the terminal cost which will be estimated as the solution to the Riccati equation. The solver will solve the entire trajectory, select the first optimal control input, move to the optimal state, and repeat the process until the quadrotor reaches the goal state.

The Augmented Lagrangian Trajectory Optimizer (ALTRO) solver was the solver used to solve the constrained optimization problem shown in Equation 16. ALTRO is a fast solver for trajectory optimization and solves problems with inequality and equality constraints as well as nonlinear dynamics [8]. ALTRO displays high performance compared to other optimization solvers like OSQP, and more details are available in Reference 8. OSQP was the other solver studied to solve Equation 16, and this solver uses the ADMM algorithm throughout a step size [10]. For this solver to work, the optimization problem is modeled as a QP problem. Details on this algorithm are shown in Reference 10. ALTRO was used in this project due to the fast solve time over OSQP and these results are shown in Reference 11.

$$\begin{aligned}
\min_{x_{1:H}, u_{1:H-1}} \quad & \sum_{n=1}^{H-1} \frac{1}{2} x_k^T Q_k x_k + \frac{1}{2} u_k^T R_k u_k + x_H^T P_H x_H \\
\text{s.t. } \quad & x_{k+1} = A_k x_k + B_k u_k \\
& u_{min} \leq u_k \leq u_{max}
\end{aligned} \tag{16}$$

D. Communication/Hardware

An NVIDIA Jetson Nano was used as the companion computer on the quadrotor and it handles all the controller scripts on board. The flight controller used was an Arduino Uno as it can efficiently send PWM signals to the motors. A Holybro X-500 kit was the drone used and the model parameters were calculated based off of the drone dimensions and hardware. By modeling the drone on Solidworks with accurate mass properties, the inertia matrix was found, and performance data from APC propellers was used to find the thrust and moment coefficients. Four Holybro 800 KV motors were the main sources of thrust, and a 14.8 V 5000 mAh battery was used to power the entire system. The thrust coefficient was found to be 0.11 and the moment coefficient was 0.044. However, these values were scaled up by a factor of 10 to allow the simulation to run. The voltage of the battery was measured using an analog pin on the Arduino, and this value will be a factor in the PWM signals that are sent to the drone to keep a hover position. The drone undergoes a calibration script that sets the maximum and minimum PWM signals to 1832 and 1148 which will correspond to the constraint set on the MPC controller. The quadrotor used in the project is shown in Figure 6.

A publisher subscriber pattern was used to communicate between various components of the system. Figure 5 shows all the nodes in the system along with the corresponding hardware. The goal was to pass the controller output to the Arduino efficiently. A networking library called Zero MQ was used to bridge different languages together via sockets and the library also provides asynchronous message processing tasks. The controller was developed in the Julia programming language which is a dynamic language for scientific and numerical computing. This script will subscribe to a state publisher and determined the control actions needed to keep the drone at a hover position. For the Julia simulation, the future drone positions were simulated using the Runge-Kutta 4th order iteration method. The optimal control actions are then published to a C++ script. The main function of the C++ script is to process the information and publish the four motor commands via serial. The boost serial library was used to transfer data between the NVIDIA Jetson Nano and the Arduino Uno which commanded the motors to move at the desired PWM signal. The data transmitted between Julia and C++ is serialized using protocol buffers for efficient data transfer. Protocol buffers are Google's method of serializing structured data via any language.

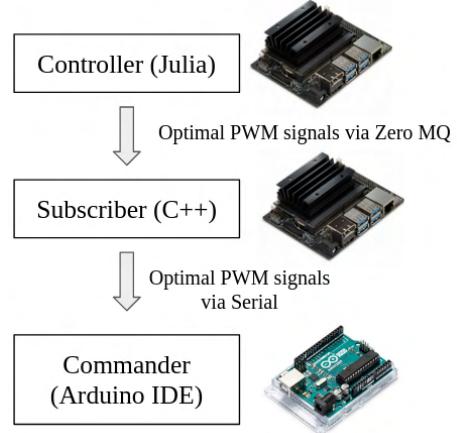


Fig. 5. Quadrotor Control Components



Fig. 6. Quadrotor used for Communication Testing

III. RESULTS

A. LQR Controller and MPC Controller

For both controller simulations, the quadrotor was set at an initial position of (2,2,2) and the reference position was set to (0,0,1). The initial orientation was also offset by 45 degrees on each axis to show the robustness of each controller. Figure 7 shows the drone position along the x axis versus time. The drone reaches the target position in the x position in 5 seconds with the LQR controller and does not show significant overshoot between 0-2 seconds. With the MPC controller, the drone reaches the reference position within 2 seconds which shows the effectiveness of the controller. For the y position (shown in 8), the drone undergoes an overshoot between 2-3 seconds with the LQR controller and then reaches the goal state within the 5 seconds. The MPC controller reaches the reference position within 2-3 seconds and does not experience significant overshoot while moving. The z position of the drone with the LQR controller converges quickly to the goal state despite the offset starting angle (shown in 9). However, MPC experiences high overshoot, yet reaches the goal state within 3 seconds. Overall the quadrotor successfully reached the reference position with both controllers despite the offset starting angle in a discrete time frame. However, MPC converged faster by 2-3 seconds in the X and Y positions which can be significant time for longer trajectories. Other start positions were also tested and

provided the same results. The Q and R matrices for LQR can be tuned to allow for aggressive control depending on the quadrotor application. However, MPC is a safer method of control as constraints are followed.

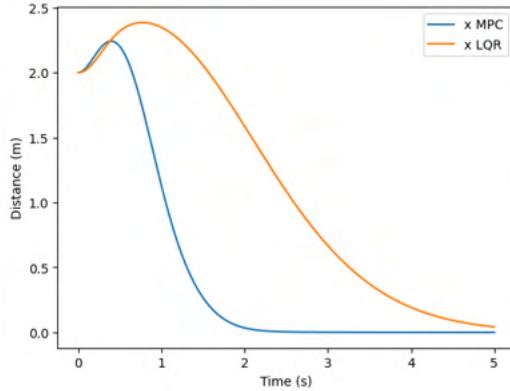


Fig. 7. Quadrotor X Position versus Time

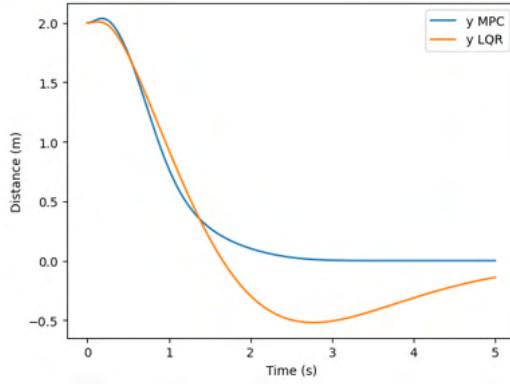


Fig. 8. Quadrotor Y Position versus Time

A video of the simulation can be accessed using the following link:

<https://drive.google.com/file/d/1sMGBAzQvgYxA7d0PvE9dTKNF-YO2u-p/view?usp=sharing>

IV. CONCLUSION AND FUTURE WORK

This work presents the results of a linear quadratic regulator and a model predictive controller on a quadrotor via simulation. The dynamics and quaternion representation were detailed to show the linearization done to solve the problem and the methods towards modeling a quadrotor. Promising results were shown with the LQR controller as it provided optimal control inputs for the quadrotor to achieve the goal state. The model predictive controller also generated safe trajectories while following torque limit constraints. A communication framework was also developed to test the algorithm on the quadrotor for future work.

Future work consists of implementing the controller on actual hardware. Additional constraints such as a limit in the z position will also be implemented to avoid the drone position to be near the ground. Another topic that will be

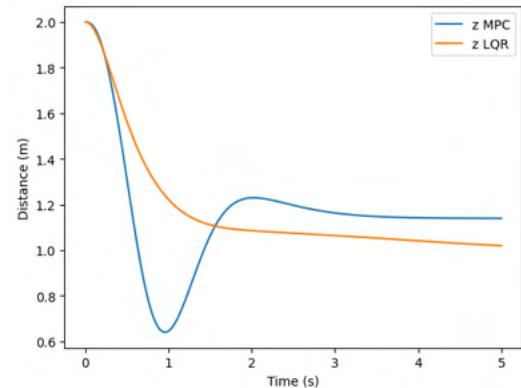


Fig. 9. Quadrotor Z Position versus Time

addressed in the future is establishing communication with a state estimator node. This will allow the drone to run the LQR/MPC controller on board and allow the controller to find an optimal control action as soon as a new state is read. The system will then be tuned by the Q and R matrices based on the performance on real hardware testing.

ACKNOWLEDGMENT

Thank you to my mentor Dr. Manchester for the guidance and suggestions throughout the project. The authors would also like to thank Ms. Burcin and Dr. Dolan for their support coordinating RISS. I'm also thankful for the members of the Robot Exploration Laboratory for their help. This work is supported by The Robotics Institute, Carnegie Mellon University.

REFERENCES

- [1] M. M. S. N. L. B. S. Chowdhury, A. Emelogu, "Drones for disaster response and relief operations: A continuous approximation model," in *International Journal of Production Economics*, 2nd ed., 2017, vol. Volume 188, pp. 167–184.
- [2] C. S. J. Scott, *Drone Delivery Models for Healthcare*, 2017.
- [3] B. D. U. Mogili, *Review on Application of Drone Systems in Precision Agriculture*. Procedia Computer Science, 2018, vol. Volume 133, pp. 502–509.
- [4] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," 2011.
- [5] K. V. Mellinger D, Michael N, "Trajectory generation and control for precise aggressive maneuvers with quadrotors."
- [6] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," pp. 153–158, 2007.
- [7] H. Kim, D. Shim, and S. Sastry, "Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles," pp. 3576–3581 vol.5, 2002.
- [8] B. E. Jackson, T. Punnoose, D. Neamati, K. Tracy, R. Jitosho, and Z. Manchester, "Altrō-c: A fast solver for conic model-predictive control," p. 8, 2021.
- [9] B. E. Jackson, K. Tracy, and Z. Manchester, "Planning with attitude." *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5658–5664, 2021.
- [10] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: an operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020. [Online]. Available: <https://doi.org/10.1007/s12532-020-00179-2>
- [11] T. A. Howell, B. E. Jackson, and Z. Manchester, "Altrō: A fast solver for constrained trajectory optimization," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 7674–7679.

Semantic Segmentation in Complex Scenes for Robotics Navigation

Zhanxin Wu¹, Xinjie Yao² and Jean Oh³

Abstract— Semantic image segmentation has been an important technique for the safe navigation of a robot in complex scenes. It is a difficult task due to the complex environment including uneven lands, indefinite boundaries, and irregular features. The complicated environmental condition makes the scene hard to understand for the robot. Most image semantic segmentation models require extremely large computing resources to reach satisfying results including high precision and short inference time. Besides, for some extreme images affected by their surroundings, such as brightness, the segmentation results may become awful in current models. To overcome these issues, this paper proposes a lightweight encoder-decoder framework for semantic segmentation to (i) recover information from uninformative pixels and (ii) project 3D LiDAR point cloud into the 2D plane to improve semantic understanding of scenes.

Index Terms— Semantic Segmentation, 3D LiDAR, RGB imagery, Extreme Image.

I. INTRODUCTION

Recent years have witnessed huge progress in autonomous driving. The autonomous driving environment becomes complicated and diverse, shown in Figure 1. In general, autonomous driving is divided into two categories, on-road and off-road. In previous years, autonomous on-road driving has received significant attention in terms of datasets for segmentation. Besides, due to the wide range of applications of autonomous off-road driving, such as exploration and rescue, there is more and more attention on autonomous off-road driving. Compared to the on-road driving environment, autonomous off-road driving is always viewed as a hard task due to its unstructured environment. Most researchers study the on-road and off-road environment separately, thus the robustness of their models is unsatisfying. Though the public scene datasets contain a mass of images, the type of images is similar. For example, Cityscapes [1] only covers urban scenes while RUGD [2] only includes the unstructured environment. Therefore, due to the limit of datasets, most models for autonomous driving only achieve high performance in a specific type of environment.

Autonomous on-road driving is mainly used in the urban environment. The structured environment including buildings and pavements makes scenes easy to understand. However, this characteristic also restricts most advanced autonomous driving models in an on-road environment. Because these

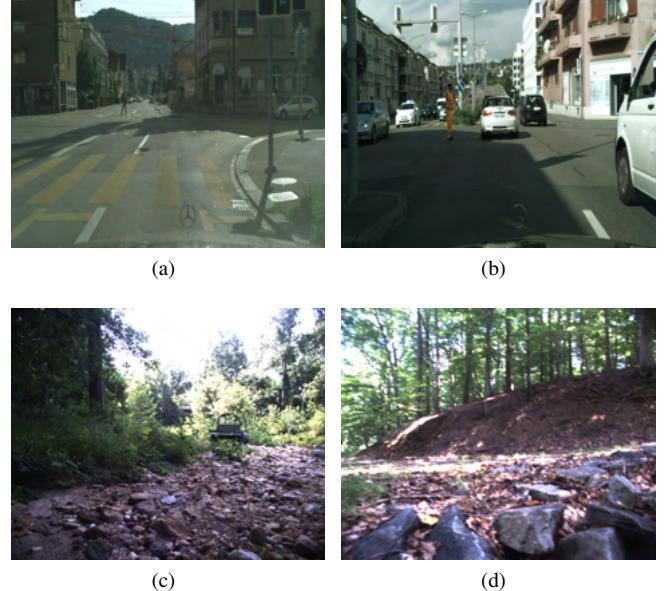


Fig. 1: (a), (b) are on-road scenes from Cityscapes [1], and (c), (d) are off-road scenes from RUGD [2].

models highly depend on lane extraction [3] and traffic sign recognition [4], which are not contained in off-road scenes. Unlike the on-road dataset including detailed classes, such as signboard, the feature class in each off-road dataset is much less. This allows research to pool classes into several regions, such as sky, traversable and non-traversable. By grouping all the classes, the low pixel density resulting in class imbalance can be solved.

In the real world, the scenes are not simple. The real environment is highly possible mixed with the on-road environment and off-road environment. To solve this issue, [5] introduced a transfer learning framework with semantic segmentation for off-road environments. This approach achieves good performance on obstacles, grass, road, and trees, which are crucial in path planning.

The extreme images, in reality, are another challenge in autonomous driving. Extreme images are defined as RGB images with enormous uninformative pixels. These pixels affected by surroundings such as light are too bright or too dark to provide information. The lack of data causes the misunderstanding of scenes and thus leads to awful performance in semantic segmentation results and interfere with path planning.

To solve the above problems, this paper proposes a lightweight semantic segmentation module with high preci-

¹Zhanxin Wu is with the School of Data Science, The Chinese University of Hong Kong, Shenzhen, Shenzhen, Guangdong, China, zhanxinwu@link.cuhk.edu.cn

²Xinjie Yao is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA., USA, xinjieyao@andrew.cmu.edu

³Jean Oh is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA., USA, jeanoh@nrec.ri.cmu.edu

sion and short inference time. Besides, this paper provides an enhanced method for extreme images: (i) label bright pixels and dark pixels as uninformative at first. (ii) classify these pixels based on their nearby classes to recover information. Then, better segmentation results can be achieved. Meanwhile, with the assistance of the point clouds from 3D LiDAR, the color error caused by the camera and environment could be minimized. Finally, the traversability of each area can be confirmed.

The remainder of this paper is organized as follows. A review of related literature is provided in section II. Section III introduces the technical approach of the problem. Section IV details the evaluation metric this paper used and experiment results compared with commonly used models. Finally, a summary and suggestions for future research are provided in section V.

II. RELATED WORK

A. Assumption-based models

Assumption-based models are named by assuming the color and boundaries of a target region. Due to the rough assumption, the segmentation results are always coarse-grained. For each pixel, the label is binary. All the assumption-based models can be further divided into rule-based and segmentation-based methods [6].

Rule-based method applies the presumed boundary and shape information of each class for segmentation. For example, the authors presented an algorithm to limit vanishing point search, which mainly relies on road edges cues [7].

Segmentation-based methods focus on specific visual characteristics of the pavement, such as color contrasts or edges in the image. Some methods utilize the structured road by fixing its shape, such as triangular [8].

B. Deep learning models

With the rapid development of the deep neural network, deep learning models could achieve higher precision in semantic segmentation than assumption-based models. With the help of fine-annotation datasets, such as Cityscapes, current deep learning methods could gain pixel-level semantic segmentation. However, the deep learning methods require large computing resources and storage. Due to the multiple layers in the neural network, most models' inference time is more than 1.5 seconds which makes real-time navigation impossible. To reduce inference time, most models have the encoder-decoder architecture, such as [9] and [10]. Apart from that, another challenge is the lack of varieties in the dataset. Most fine-annotated datasets are mainly for urban environments. To study complex environments including unstructured scenes, scholars have to try various methods to avoid relying on these datasets. The mainstream method is to do transform training from existing datasets, such as Cityscapes and Ade20k [11]. Another main idea is to ask other synthetic data for help, such as 3D LiDAR data [12] or audio data [13] to achieve the height and distance of objects in the environment.

III. METHODOLOGY

A. Architecture

The whole architecture is represented in Figure 2. This paper consists of two primary streams, RGB image and LiDAR point cloud data [14]. The RGB images are input into the semantic segmentation model and finally, become a pixel-wise labeling result. Meanwhile, the 3D LiDAR point clouds would be projected into the 2D plane. According to the height of each object in the point clouds, we could predict whether it is traversable or not. Therefore, we could gain a 2D binary map including traversable and non-traversable regions from the point clouds. Then, the pixel-wise segmentation results and the projected 2D plane should be combined to predict the traversable region for robotics. The result is a binary map showing traversable region continuously updated estimates of relevant traversable information for navigation.

All computers run Ubuntu Linux. The codes are implemented in Python, using CUDA to make effective use of the GPU(Tesla T4).

B. First stream: Semantic Segmentation on Images

1) *Encoders and Decoders*: The Xception [15] architecture is a linear stack of depthwise separable convolution layers with residual connections. It has the same number of parameters as Inception V3 but improves performance by efficiently using model parameters. In [16], MobileNetv2 was introduced to mobile device. The main purpose of MobileNetv2 is to solve computer vision tasks with as few parameters as possible. Researchers are allowed to ensure high accuracy and precision with limited resources. In [17], the author proposed DeepLabv3 holding abundant semantic information from the encoder module. In the encoder-decoder framework, the decoder module recovers class edge information, and the encoder module extract features by applying convolution.

2) *Tricks on Extreme Images*: In the public datasets including both on-road and off-road datasets, the images always have appropriate brightness, contrast, etc. However, in reality, imagery from a camera could have extremely low brightness or high brightness, due to the effect of surroundings. For example, in the days of strong sunlight, imagery has plenty of pixels with high brightness and thus contains little information. To deal with this issue, this paper proposes a trick: (i) label extreme pixels as uninformative, and a mask full of uninformative pixels. (ii) lock each pixel in the mask with a rectangular block to make sure the pixel is in the center of the block. (iii) calculate the most frequent class in each block and provide it to the central pixel. (iv) combine the results with the previous image and gain the final product. Finally, we could retrieve information on uninformative pixels.

C. Second Stream: 3D LiDAR Point Cloud Projection

From 3D LiDAR, a point cloud can be achieved. To gain a 2D view of a scene, each 3D point should be projected into the image plane using a perspective transformation and thus the 3D point could become the matching pixel p in the

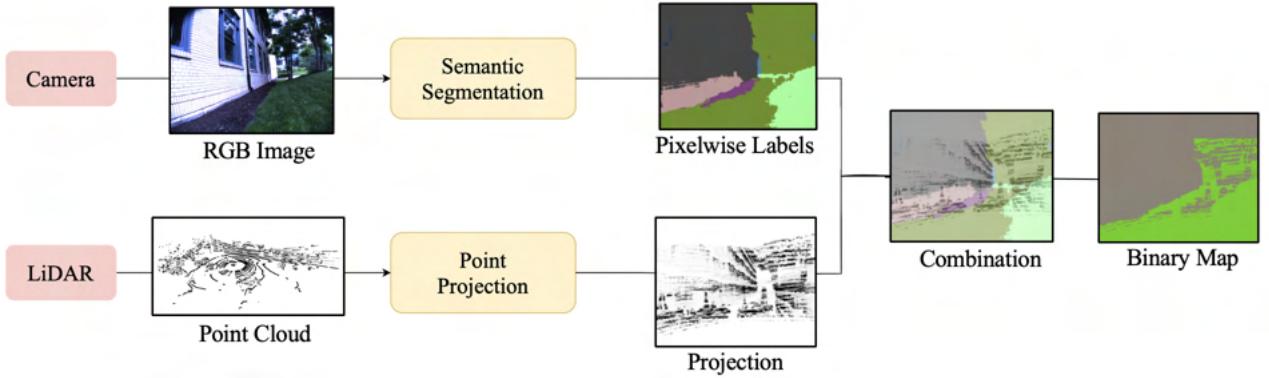


Fig. 2: Overview of the Model. Note: First stream is RGB images from the camera and Second stream is point clouds from the 3D LiDAR

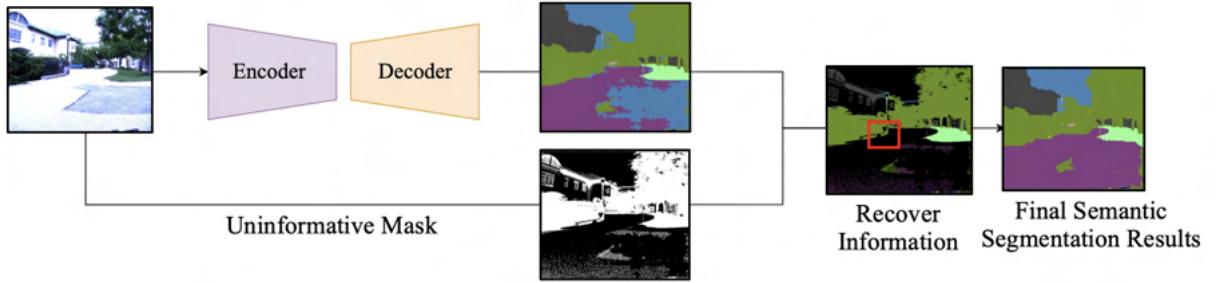


Fig. 3: Semantic Segmentation Framework including tricks on extreme images (uninformative mask)

2D image plane. Both the 3D point and its pixel point to the homogeneous location in reality. From the 3D LiDAR point cloud, this model gets the height of each point, and then views all the areas that are above the threshold, such as 30 centimeters, as non-traversable. Finally, this model can attain a 2D binary map including the traversable region and the non-traversable region.

D. Streams Combination

The outputs for semantic segmentation and 3D LiDAR projection are both pixel-wise. The purpose to combine these two maps is to get more clear edges for unstructured classes, such as terrain and vegetation. Therefore, for the structured class, we trust the most frequent label in the point cloud. For example, if most pixels in the projected 2D plane are traversable, we will view this whole area as a traversable region. Through this rule, we maintain the shape of structured classes and keep their edge information. For the unstructured class, we highly depend on the point cloud, so that the shape of such classes in the final map could be changed and become discrete. In this case, we better collect unstructured classes' shapes. Through the combination method, the model not only understands the scene through color in imagery but also the height of each object. In this case, the color error caused by the environment and the camera could be minimized, which could help better path planning for robotics navigation.

IV. EVALUATION

In this section, the model has experimented on two datasets, Cityscapes, and our campus frame. The evaluation of image semantic segmentation of Cityscapes and our campus frames was done using different combinations of encoders and decoders. The model is trained on the Cityscapes train set to gain pre-trained weights.

A. Segmentation Precision

The results of the model are shown in Table I. The mean IoU for the datasets can be calculated by:

$$IoU = \frac{1}{n} \sum_{n=1}^n \frac{(TP)_n}{(TP)_n + (FP)_n + (FN)_n} \quad (1)$$

where n is the number of classes. TP represents True Positive, and FP represents False Positive, and FN represents False Negative.

TABLE I: Semantic Segmentation Results on Cityscapes

Decoder	Encoder	mIoU(%)
	MobileNetv2	70.71
DeeplabV3	Xception65	78.79
	Xception71	80.31

Figure 4 and Tabel II show the effect of information retrieval on extreme images. In Figure 4(a), the road contains exceedingly bright pixels, so that part of the road is classified as the sky. According to the uninformative mask mentioned

TABLE II: Semantic Segmentation Results on Campus Frame

Metric(%)	road mIoU	building mIoU	terrain mIoU	sky mIoU	mean totalclass mIoU	mean pixel accuracy
Original Results	59.12	83.35	46.69	36.44	63.13	76.36
Enhanced Results	87.76	83.64	47.11	88.73	77.45	89.24

in Figure 3, the information of these extreme pixels can be recovered. Thanks to the method, the mIoU of road class increased from 0.59 to 0.87. Besides, the mean pixel accuracy could achieve 0.89. The semantic segmentation accuracy becomes satisfying.

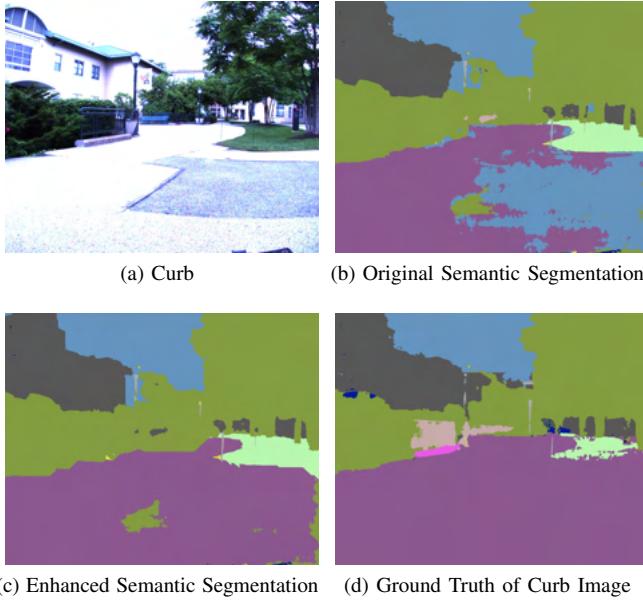


Fig. 4: Enhanced Semantic Segmentation Results on Extreme Images.

B. Inference time

The inference speed of the model was tested on Tesla T4. The input resolution of campus frames is 1280*1024. Table III shows the inference speed of different models. This distinguishes the performance of encoders and decoders.

TABLE III: Inference Time on Campus Frame

Decoder	Encoder	Inference Time(ms)
	MobileNet2	276
DeeplabV3	Xception65	5087
	Xception71	592

According to Table I and Table III, even though the mIoU and pixel accuracy for Xception65 are high, the price is a long inference time. It costs almost ten times Xception71's inference time. For each image, the time needed for Xception65 is 5 seconds, which makes it impossible for robotics to do real-time path planning. The performance of MobileNetv2 and Xception71 are acceptable. However, the precision for lightweight MobileNetv2 is too low to be applied in reality. Therefore, the DeeplabV3 and Xception71 become the best choice among these three models.

C. Combination Results

Based on 3D LiDAR data and RGB imagery, we could obtain two images with binary labels, traversable and non-traversable regions. Since RGB imagery could be affected by surroundings, and there is an inevitable error in the semantic segmentation model, it is hard to only depend on semantic segmentation for navigation. The 3D LiDAR data could assist path planning by providing the height of each pixel. In this model, we combine height gained from 3D LiDAR and semantic segmentation results to achieve a final traversable path. Figure 5 shows the final results. If the model only considers RGB images or 3D LiDAR point cloud, the pixel accuracy is 0.7195 and 0.4652 respectively. However, the framework proposed in this paper could increase the pixel accuracy to 0.8754, because it makes good use of both images and points clouds.

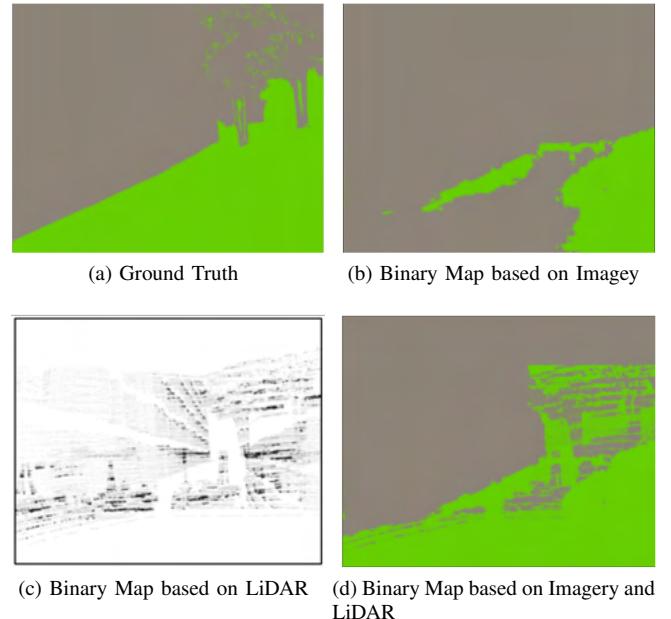


Fig. 5: Results using Different Information (Note: For segmentation results, grey represents non-traversable and green represents traversable. For point cloud, black represent non-traversable and white represent traversable)

V. CONCLUSION

This paper has presented a semantic segmentation model on two datasets. This model achieves good performance not only in mIoU but also in inference time. The traversable region obtained from the model could assist robotics to understand complex scenes and thus contribute to path planning

for navigation. This model could be extended to extreme images with a high density of uninformative pixels. With the help of RGB imagery and 3D LiDAR data, the high accuracy of predicting traversable regions can be promised.

To progress closer towards complex scene understanding, some steps can be taken to further improve the model performance. First, we could study the robustness of the approach under different climatic conditions changes [18]. Because the color of vegetation class varies in different seasons, the model that is only trained on the scenes collected for a short period needs to be proved its robustness in all the conditions. Second, current 3D LiDAR data point clouds are sparse. In this case, some obstacles like rocks may be ignored. Therefore, denser point clouds could be collected in the future and better help path planning.

ACKNOWLEDGMENT

The work is supported by the BIG Lab at the Robotics Institute, Carnegie Mellon University. We would also like to give our special thanks to Rachel Burcin, Dr. John Dolan, and all the members of the Robotics Institute Summer Scholars (RISS) program for making this amazing program possible.

REFERENCES

- [1] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.
- [2] M. Wigness, S. Eum, J. G. Rogers, D. Han, and H. Kwon, “A rug dataset for autonomous navigation and visual perception in unstructured outdoor environments,” in *International Conference on Intelligent Robots and Systems (IROS)*, 2019.
- [3] F. Zhang, H. Stähle, C. Chen, C. Buckl, and A. Knoll, “A lane marking extraction approach based on random finite set statistics,” in *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 1143–1148.
- [4] A. Mogelmose, M. M. Trivedi, and T. B. Moeslund, “Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1484–1497, 2012.
- [5] S. Sharma, J. E. Ball, B. Tang, D. W. Carruth, M. Doude, and M. A. Islam, “Semantic segmentation with transfer learning for off-road autonomous driving,” *Sensors*, vol. 19, no. 11, p. 2577, 2019.
- [6] B. Gao, S. Hu, X. Zhao, and H. Zhao, “Fine-grained off-road semantic segmentation and mapping via contrastive learning,” *arXiv preprint arXiv:2103.03651*, 2021.
- [7] J. Shi, J. Wang, and F. Fu, “Fast and robust vanishing point detection for unstructured road following,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 970–979, 2015.
- [8] S. Zhou and K. Iagnemma, “Self-supervised learning method for unstructured road detection using fuzzy support vector machines,” in *2010 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2010, pp. 1183–1189.
- [9] Y. Wang, Q. Zhou, J. Liu, J. Xiong, G. Gao, X. Wu, and L. J. Latecki, “Lednet: A lightweight encoder-decoder network for real-time semantic segmentation,” in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 1860–1864.
- [10] J. Jiang, L. Zheng, F. Luo, and Z. Zhang, “Rednet: Residual encoder-decoder network for indoor rgb-d semantic segmentation,” *arXiv preprint arXiv:1806.01054*, 2018.
- [11] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, “Scene parsing through ade20k dataset,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 633–641.
- [12] Y. Wang, T. Shi, P. Yun, L. Tai, and M. Liu, “Pointseg: Real-time semantic segmentation based on 3d lidar point cloud,” *arXiv preprint arXiv:1807.06288*, 2018.
- [13] J. Zürn, W. Burgard, and A. Valada, “Self-supervised visual terrain classification from unsupervised acoustic feature learning,” *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 466–481, 2020.
- [14] D. Maturana, P.-W. Chou, M. Uenoyama, and S. Scherer, “Real-time semantic mapping for autonomous off-road navigation,” in *Field and Service Robotics*. Springer, 2018, pp. 335–350.
- [15] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” 07 2017, pp. 1800–1807.
- [16] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.
- [17] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” *arXiv preprint arXiv:1706.05587*, 2017.
- [18] K. Viswanath, K. Singh, P. Jiang, S. Saripalli, et al., “Offseg: A semantic segmentation framework for off-road driving,” *arXiv preprint arXiv:2103.12417*, 2021.
- [19] A. Vemula, K. Muelling, and J. Oh, “Social Attention: Modeling Attention in Human Crowds,” in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 4601–4607.
- [20] P. Jiang, P. Osteen, M. Wigness, and S. Saripalli, “Rellis-3d dataset: Data, benchmarks and analysis,” *arXiv preprint arXiv:2011.12954*, 2020.
- [21] R. P. K. Poudel, S. Liwicki, and R. Cipolla, “Fast-scnn: Fast semantic segmentation network,” 2019, cite arxiv:1902.04502. [Online]. Available: <http://arxiv.org/abs/1902.04502>
- [22] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, “Encoder-decoder with atrous separable convolution for semantic image segmentation,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.
- [23] H. Jeong, Y. Oh, J.-H. Park, B. Koo, and S. W. Lee, “Vision-based adaptive and recursive tracking of unpaved roads,” *Pattern Recognition Letters*, vol. 23, no. 1-3, pp. 73–82, 2002.
- [24] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” *arXiv preprint arXiv:1511.07122*, 2015.
- [25] A. Valada, G. L. Oliveira, T. Brox, and W. Burgard, “Deep multispectral semantic scene understanding of forested environments using multimodal fusion,” in *International symposium on experimental robotics*. Springer, 2016, pp. 465–477.

Concept Whitening for Interpretability in Multi Agent Reinforcement Learning

Renos Zabounidis¹, Joseph Campbell², Ini Oguntola², Dana Hughes², and Katia Sycara²

Abstract—Deep Multi Agent Inverse Reinforcement learning (MAIRL) has imbued artificial agents with the capability of understanding and adapting to human behavior. However, the black box, data driven nature of these agents poses an obstacle to interpretability. Posthoc analysis of neural networks is often insufficient, either requiring extraneous assumptions or surveying humans. Concept whitening a data driven technique developed for computer vision neural network interpretability. In this paper, we propose an extension of Concept Whitening for MAIRL. We implement our algorithm on top of several state of the art MAIRL models including CoDAIRL, GAIL, and AIRL. Models including concept whitening are shown to retain similar levels of performance to their counterparts, while increasing interpretability.

Index Terms—keywords, choose from
<https://www.ieee-ras.org/publications/ra-l/keywords>

I. INTRODUCTION

Real world multi agent autonomous systems require not only the ability of an agent to display precise control and decision making in complex environments, but the ability to compete and coordinate with other agents. Learning optimal behavior for each agent is especially difficult to the intricacies arising from the interplay between agents. Capturing these eccentricities in hard coded, rule based agents has proved to be cumbersome and in most cases impossible. Success has been found in data driven algorithms, which use self play or expert demonstrations to learn optimal behavior. The most common framework used to learn optimal behavior in complex environments has been Deep Reinforcement Learning.

Deep Reinforcement Learning has enabled agents to achieve superhuman performance in many tasks, including real world robotics [1] [2] and games [3] [4] [5]. These successes directly result in most Multi Agent RL Algorithms (MARL) being direct analogues of their single agent counterparts [6] [7]. The sieve solution is to directly model the joint policy of all the agents. However, as the joint action space scales exponentially with the number of agents, decentralized execution and centralized training algorithms have been created in order to make training computationally feasible. In doing so, MARL algorithms have to deal with three additional challenges: Nonstationarity, shadowed equilibrium, and the credit assignment problem.

¹Renos Zabounidis is an Undergraduate at the University of Massachusetts Amherst, rzbounidis@cs.umass.edu

²J. Campbell, I. Oguntola, D. Hughes, and K. Sycara are with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA {jacampbe, ioguntol, danahugh, katia}@cs.cmu.edu

The nonstationarity problem results from the fact that the transition function of an MARL agent constantly changes due to the changing policies of the other agents. The shadowed equilibrium problem occurs in a prisoners dilemma style scenario where a suboptimal nash equilibrium in the joint-policy space is preferred in training over the optimal policy. The credit assignment problem occurs when rewards are shared among cooperative agents. Since multiple agents contribute towards the gaining of a reward, the problem exists of who to credit with the gain of that reward.

In solving these problems, especially in a partially observable setting, MARL agents optimally would model the beliefs, desires, and intentions of other agents. Yet it is not relatively obvious how successful agents are at this task, as no explicit mental state of other agents is learned in training. Instead, agents must learn to extract the beliefs, desires, and intentions (BDI) of other agents from the state of the world.

This implicit modeling is a problem, especially for tasks which contain people, who contain extremely high variability with respect to BDI. In such cases, it is not enough to model a human as another environmental dynamic, as the humans responses to the agents actions becomes central to any optimal policy for the agent. Additionally, an implicit, black box BDI model is not desirable in real world, high stakes scenarios where humans being able to understand the beliefs of autonomous agents they are working with is essential to building trust.

In this work, we enable existing MARL and MAIRL agents to learn a limited theory of mind. We do this by applying Concept Whitening, a neural network module introduced by Chen et Al [8] meant to increase interpretability. By learning concepts which correspond to beliefs within a theory of mind, we can both make existing MARL MAIRL methods more interpretable and provide guarantees as to which beliefs the network is able to learn.

II. RELATED WORK

Many papers in Multi Agent Reinforcement Learning attempt to extend a single agent algorithm to the Multi Agent domain [6] [7]. Two main approaches currently exist: distributed and centralized training [9]. Under distributed training, no explicit information or parameters are shared between agents during training. In practice, this limitation has been shown to have inferior performance compared to centralized training [10].

Under centralized training, agents can explicitly exchange information during training, but not during testing [11]. This approach has yielded state of the art MARL models [12]

[13] [14]. These models state of the art performance can be attributed to the extra information given to them at training time [14]. For example, [14] learns an additional neural network which predicts opponents policies given the current joint state. These networks allow agents to more effectively model each others policies. However, the extent to which this modeling is successful remains unknown to a human onlooker, due to the fact the use of deep neural networks makes the problem uninterpretable. Current literature in Multi Agent RL has not explored the effect of interpretability on the learned models, and whether this is capable of impacting performance.

Interpretability has been most actively studied in the field of computer vision, in which there are two main schools of thought: ad hoc and post hoc models. Concept whitening is a method in the latter school, along with This Looks Like That (TLLT) [15]. TLLT uses saliency maps to learn prototypical images in each class. It is therefore not suited to RL, where no classifier is used. Other posthoc models exist for neural networks; however, these networks are not optimal, as their pos hoc nature force the models to make assumptions, rather then constraints, on the latent spaces they try to explain [8].

III. PRELIMINARIES

A. Markov Games

A Markov game (MG) [16] is defined as an extension of a Markov Decision Process (MDP). An MG of N agents is defined as a tuple $\langle N, S, A^1, \dots, A^n, P, r^1, \dots, r^n, p_0, \gamma \rangle$ with S being the set of states, each A^i and $r^i : S \times A^1 \times \dots \times A^N \rightarrow \mathbb{R}$ corresponding to the action space and reward function of the i th agent, $P : S \times A^1 \times A^N \times S \rightarrow [0, 1]$ is the state transition probability distribution, p_0 is the probability distribution of the initial state s^0 , and $\gamma \in [0, 1]$ is the discounted factor.

Let $-i$ denote the set of agents except for i . For example, (a_i, a_{-1}) represents (a_1, \dots, a_n) . π denotes the joint policy, r denotes the (r_1, \dots, r_n) , and a denotes (a_1, \dots, a_n) . Expectation with respect to a policy π denotes taking an expectation with respect to the trajectories it generates if the expectation is taken over a state. Otherwise the policy only samples over the next step-action a .

B. ϵ -Nash Equilibrium

An ϵ -Nash Equilibrium is a relaxed version of a nash equilibrium [17] [14]. We define a $\epsilon - NE$ as a strategy profile (π_*^i, π_*^{-i}) such that $\exists \epsilon > 0$. s.t.

$$v^{(i)}(s, \pi_*^{(i)}, \pi_*^{(-i)}) \geq v^{(i)}(s, \pi^{(i)}, \pi_*^{(-i)}) - \epsilon, \forall \pi^{(i)} \in \Pi^{(i)}$$

where $v^{(i)} = \mathbb{E}_{\pi^{(i)}, \pi^{(-i)}, s_0=s} [r^{(i)}(s_t, a_t^{(i)}, a_t^{(-i)})]$ is the value function of agent i under state s , and $\Pi^{(i)}$ is the set of all possible policies of agent i . $\epsilon - NE$ is strictly weaker then NE, since every NE is equivalent to an $\epsilon - NE$ when $\epsilon = 0$.

C. Multi-agent actor-critic with Kronecker factor (MACK)

MACK is a multi agent version of Multi-agent actor-critic with Kronecker factor Trust Region without the trust region [6] [18]. MACK uses centralized training with decentralized execution, a framework in which policies are trained with centralized information, but that information is not given during test time [19]. Under this paradigm, the advantage function for an agent is defined as a function of all agents observations and actions:

$$A_{\phi_i}^{\pi_i}(S, a_t) = \sum_{j=0}^{k-1} (\gamma^j r(s_{t+j}, a_{t+j}) + \gamma^k V_{\phi_i}^{\pi_i}(s_{t+k}, a_{-i,t})) V_{\phi_i}^{\pi_i}(s_t, a_{-i,t}) \quad (1)$$

where $V_{\phi_i}^{\pi_i}(s_k, a_{-i})$ is the baseline for i , utilizing the additional information for variance reduction. Approximated neural policy gradients are used to optimize both ϕ and θ with a linear decay learning rate schedule.

D. Concept Whitening

Concept whitening consists of two parts: whitening and orthogonal transformation [8]. Whitening is achieved using iterative normalization [20] and the orthogonal transformation is iteratively learned it by gradient methods on the Stiefel manifold as defined in [8].

Let $Z_{d \times n}$ be the output of a layer before concept whitening, where n denotes the batch size and each column $z_i \in \mathbb{R}^d$ contains the latent space of the i th sample in the batch.

The whitening transformation decorrelates and standardizes the data:

$$\psi(Z) = W(Z - \mu \cdot 1_{n \times 1}^T) \quad (2)$$

where μ is the batch mean and W is the whitening matrix such that $W^T W = \Sigma^{-1}$, where $\Sigma = \frac{1}{n}(Z - \mu 1^T)(Z - \mu 1^T)^T$ is the covariance matrix.

Whitening matrices remain invariant under orthogonal transformation, and thus $W' = QW$ will satisfy equation III-D. Concept whitening takes advantage of this property to learn Q^T such that for each concept in the output $\psi(Z)$ is activated alone one column. Formally, this means we need to find an orthogonal matrix Q such that for every concept i , the i th column of Q is optimized in the following fashion:

$$\begin{aligned} \max_{q, q_2, \dots, q_k} \sum_{i=1}^k \frac{1}{n_i} q_i^T \psi(Z_{c_j}) \cdot 1_{n_j \times 1} \\ \text{s.t. } Q^T Q = I_d \end{aligned} \quad (3)$$

The forward pass of the concept whitening module is displayed in Algorithm 2.

IV. METHODOLOGY

A. Conceptual Theory of Mind "Choosing Concepts"

Here we list a methodology for choosing appropriate concepts. Concepts which are useful should not be able to be directly learned from the world state, as that is given to the agent at every timestep. Instead, aligning with theory of

Algorithm 1 Forward Pass of CW Module

Input: mini-batch input $Z \in \mathbb{R}^{d \times m}$, $Q \in \mathbb{R}^{d \times d}$ (Rotation Matrix), momentum parameter $\mathcal{B} \in [0, 1]$

Output: concept whitened representation $\hat{Z} \in \mathbb{R}^{d \times m}$

- 1: **Optimization Variables:** Running ZCA whitening matrix $W \in \mathbb{R}^{d \times d}$, Running Batch Mean $\mu \in \mathbb{R}^d$
- 2: calculate batch mean $\mu = \frac{1}{m}Z \cdot 1$. If test time, use running batch mean μ
- 3: Calculate center of activation: $Z_C = Z - \mu \cot 1^T$
- 4: Calculate ZCA-whitening Matrix W (Algorithm 1 of [?], see appendix for convenience)
- 5: Calculate Concept whitened representation: $\hat{Z} = Q^T \cdot W \cdot Z_C$
- 6: **if** Training **then**
- 7: Update running_mean \mathcal{V} as $\mathcal{B} \cdot \mu + (1 - \mathcal{B}) \cdot \mathcal{V}$ and
 Update running_rotation matrix \mathcal{W} as $\mathcal{B} \cdot Q + (1 - \mathcal{B}) \cdot \mathcal{Q}$
- 8: **end if**
- 9: **return** P

mind, concepts should represent different verifiable beliefs the agent holds about the world. Being verifiable means that looking back at a trajectory, one can automatically label whether a concept should have been active at a certain timestep.

B. Concept Whitening MACK

Below is the algorithm for training CW MACK.

Algorithm 2 CW MACK Training

Input: num_batches, num_epochs

Output: out

Initialisation :

- 1: first statement
- 2: **for** $i = 0$ to number batches **do**
- 3: Run Model on the environment until num_batches trajectories are accumulated
- 4: take MACK training step on Model
- 5: **if** $i \bmod 20 = 0$ **then**
- 6: statement..
- 7: **for** $j = 1, \dots, \text{num_concepts}$ **do**
- 8: load concept dataset j
- 9: pass a batch of concept j through each agents model
- 10: **end for**
- 11: Update Orthogonal matrix in whitening Using algorithm based on equation 2
- 12: **end if**
- 13: **end for**
- 14: **return** P

The running mean and whitening matrix is iteratively updated during each training pass as seen in Algorithm 1. The rotation matrix stays constant until it is updated using the optimization process defined in equation 2.

V. EXPERIMENTAL SETUP

A. Particle Environments

In this paper, we use the Multi Agent particle environments, a set of competitive and cooperative environments with multiple agents.

1) *Predator-Pray*: In the Predator-Pray task, each agent learns five concepts:

- 1) Whether predator 1 will tag the pray in 5 timesteps
- 2) Whether predator 2 will tag the pray in 5 timesteps
- 3) Whether predator 5 will tag the pray in 5 timesteps
- 4) Whether the pray will hit landmark 1 in 5 timesteps
- 5) Whether the pray will hit landmark 2 in 5 timesteps



Fig. 1. An example of the Predator Pray Environment. There are two landmarks, three predators, and one pray.

VI. EXPERIMENTAL RESULTS

Currently results are limited due to the model not converging.

VII. CONCLUSION

APPENDIX

A. MODEL ARCHITECTURES

Three layer MLP networks with 128 neurons were used for policy networks for all actors used in the paper. After the first two layers, Concept Whitening modules were applied.

ACKNOWLEDGMENT

The authors would like to thank Rachel Burcin, John Dolan, and all the other members of the RI community making RISS 2021 possible and their dedication throughout the summer. This work was supported by the CMU Robotics Institute Summer Scholars Lab Scholarship. This material is based upon work supported by the National Science Foundation under Grant No. 1659774.

REFERENCES

- [1] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.
- [2] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” *arXiv preprint arXiv:1509.02971*, 2015.
- [3] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, *et al.*, “Grandmaster level in starcraft ii using multi-agent reinforcement learning,” *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

- [4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.
- [5] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," *arXiv preprint arXiv:1712.01815*, 2017.
- [6] J. Song, H. Ren, D. Sadigh, and S. Ermon, "Multi-agent generative adversarial imitation learning," *arXiv preprint arXiv:1807.09936*, 2018.
- [7] C. Yu, A. Velu, E. Vinitsky, Y. Wang, A. Bayen, and Y. Wu, "The surprising effectiveness of mappo in cooperative, multi-agent games," *arXiv preprint arXiv:2103.01955*, 2021.
- [8] Z. Chen, Y. Bei, and C. Rudin, "Concept whitening for interpretable image recognition," *Nature Machine Intelligence*, vol. 2, no. 12, pp. 772–782, 2020.
- [9] P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, vol. 33, no. 6, pp. 750–797, 2019.
- [10] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *International Conference on Autonomous Agents and Multiagent Systems*. Springer, 2017, pp. 66–83.
- [11] S. Kumar, P. Shah, D. Hakkani-Tur, and L. Heck, "Federated control with hierarchical multi-agent deep reinforcement learning," *arXiv preprint arXiv:1712.08266*, 2017.
- [12] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *arXiv preprint arXiv:1706.02275*, 2017.
- [13] S. Iqbal and F. Sha, "Actor-attention-critic for multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2961–2970.
- [14] M. Liu, M. Zhou, W. Zhang, Y. Zhuang, J. Wang, W. Liu, and Y. Yu, "Multi-agent interactions modeling with correlated policies," *arXiv preprint arXiv:2001.03415*, 2020.
- [15] C. Chen, O. Li, C. Tao, A. J. Barnett, J. Su, and C. Rudin, "This looks like that: deep learning for interpretable image recognition," *arXiv preprint arXiv:1806.10574*, 2018.
- [16] M. L. Littman, "Markov games as a framework for multi-agent reinforcement learning," in *Machine learning proceedings 1994*. Elsevier, 1994, pp. 157–163.
- [17] J. Nash, "Non-cooperative games," *Annals of mathematics*, pp. 286–295, 1951.
- [18] Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba, "Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation," *Advances in neural information processing systems*, vol. 30, pp. 5279–5288, 2017.
- [19] J. N. Foerster, Y. M. Assael, N. De Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," *arXiv preprint arXiv:1605.06676*, 2016.
- [20] L. Huang, Y. Zhou, F. Zhu, L. Liu, and L. Shao, "Iterative normalization: Beyond standardization towards efficient whitening," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4874–4883.
- [21] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Advances in neural information processing systems*, 2000, pp. 1008–1014.
- [22] M. Świechowski, K. Godlewski, B. Sawicki, and J. Mańdziuk, "Monte carlo tree search: A review of recent modifications and applications," *arXiv preprint arXiv:2103.04931*, 2021.
- [23] A. Gopnik and H. M. Wellman, "Why the child's theory of mind really is a theory," 1992.
- [24] N. Rabinowitz, F. Perbet, F. Song, C. Zhang, S. A. Eslami, and M. Botvinick, "Machine theory of mind," in *International conference on machine learning*. PMLR, 2018, pp. 4218–4227.
- [25] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Pérolat, D. Silver, and T. Graepel, "A unified game-theoretic approach to multiagent reinforcement learning," *arXiv preprint arXiv:1711.00832*, 2017.
- [26] A. Lerer and A. Peysakhovich, "Learning existing social conventions via observationally augmented self-play," in *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 2019, pp. 107–114.
- [27] W. Yoshida, R. J. Dolan, and K. J. Friston, "Game theory of mind," *PLOS Computational Biology*, vol. 4, no. 12, pp. 1–14, 12 2008. [Online]. Available: <https://doi.org/10.1371/journal.pcbi.1000254>
- [28] C. Baker, R. Saxe, and J. Tenenbaum, "Bayesian theory of mind: Modeling joint belief-desire attribution," in *Proceedings of the annual meeting of the cognitive science society*, vol. 33, no. 33, 2011.
- [29] C. L. Baker, J. Jara-Ettinger, R. Saxe, and J. B. Tenenbaum, "Rational quantitative attribution of beliefs, desires and percepts in human mentalizing," *Nature Human Behaviour*, vol. 1, no. 4, pp. 1–10, 2017.
- [30] J. Tenenbaum, "Building machines that learn and think like people," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 5–5.

Drone Camera Calibration via Neural and Heuristic Method

Longwen Zhang¹, Yaoyu Hu² and Sebastian Scherer²

Abstract— Camera calibration is an indispensable part of drone tasks, especially in tasks that need to reconstruct the colored RGB surface. In order to make the reconstructed color better match the user’s perception, white balance and vignetting reduction are mainly needed. Despite the existence of tools such as automatic white balance correction, drones in the production environment still mainly perform camera calibration by manually calibrating colors with a standard color chart before every flight to ensure color consistency during the whole flight. To this end, this work proposes a neural-network-based method to replace manual operations, which enables automatic white balance and vignetting reduction through multiple cameras with a few shot on a color chart. The proposed method can accurately detect arbitrary color charts without any prior knowledge and adopt optimization to accomplish calibration. Extensive real-world experiments demonstrate the effectiveness of our method for automatic camera calibration even using edge devices.

Index Terms— AI-Based Methods, Object Detection, Segmentation and Categorization, Camera Calibration, Vignetting Correction

I. INTRODUCTION

Drones have been widely used in reconstruction tasks all the time. Camera color calibration is always an indispensable part of drone tasks. How to calibrate the camera fast and conveniently remains unsolved.

The important reason for color and brightness correction is (1) we need the stereo cameras produce the same color and brightness on the same object observed in the scene; (2) we need the color and brightness to be consistent across different viewing angle and camera locations. These reason rules out most of the single-camera auto-white balance functionalities. For vignetting correction, it needs a dedicated and controlled environment to be able to conduct the calibration.

In this paper, we present a novel approach to achieve color calibration, as illustrated in Fig. 1. Our approach combines both neural and traditional methods to achieve color calibration and reduce the vignetting effect.

To summarize, our main contributions include:

- We present a fast neural color calibration chart recognition and localization approach, which robustly works under extreme environmental conditions.
- We propose a heuristic approach based on the traditional computer vision method to enable automatic color calibration with no prior information on the appearance

¹Longwen Zhang is with the School of Information Science and Technology, ShanghaiTech University zhanglw2@shanghaitech.edu.cn

²Yaoyu Hu, Sebastian Scherer are with Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA {yaoyuh, basti}@andrew.cmu.edu

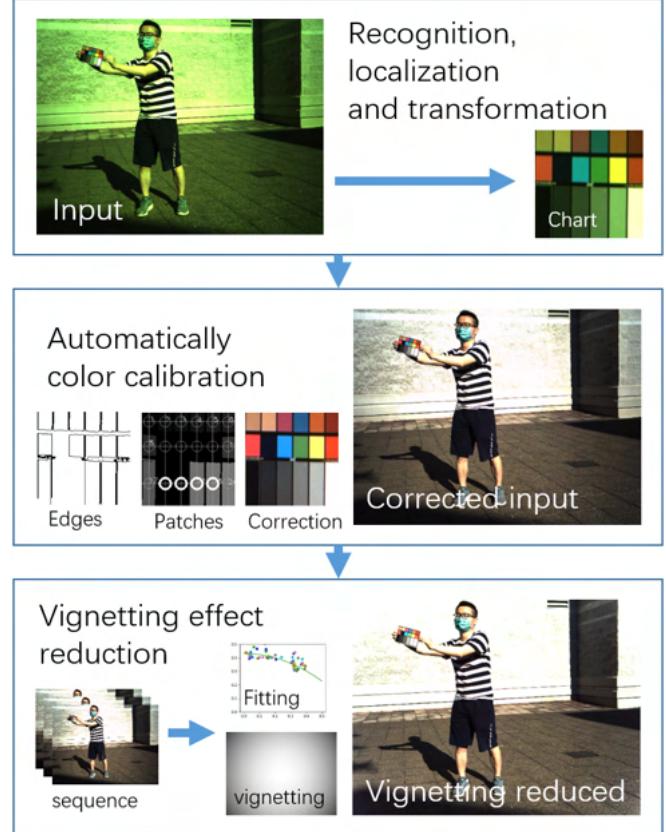


Fig. 1. Process overview.

of the color calibration chart, with only making few assumptions of the color patches pattern in various color charts.

- We further adopted a simple and novel approach to reduce the vignetting effect caused by the lens, which is inspired by [1].

II. RELATED WORKS

Automatic white balance. The gray world algorithm assumes the average of reflectance of a scene is chromatic in an image with sufficient color variations [2]. Perfect Reflector Method assumes that the brightest pixel in an image corresponds to an object point on specular surface, which reflects the true color of the light and could be used as a reference color [2]. Fuzzy Rule Method uses fuzzy logic rules for determination of the color parameters in order to minimize the color temperature difference of various light sources [3]. However, these methods could not ensure stereo



Fig. 2. Sample images from Open Images Challenge 2018/2019 test set.



Fig. 3. Sample color calibration charts.

cameras to produce the same color and brightness on the same object observed in the scene.

Vignetting effect reduction. An automatic method [4] was proposed to estimate vignetting function based on two images acquired with difference settings of lens aperture and focal length, which will change the effect of vignetting in images. An online method [1] was proposed to estimate vignetting function based on multiple frames in a sequence.

III. COLOR CHART DATASET OVERVIEW

Our goal is to robustly recognize the color calibration chart in the wild under extreme environmental conditions where images are overexposed or underexposed and with blurring and noise, and recover its position precisely. To provide ground truth supervision for color calibration chart recognition and localization, we generate a high-quality color calibration chart dataset. As illustrated in Fig. 2, we use Open Images Challenge 2018/2019 test set [5] as background to generate our dataset, which contains 99,999 images of various scenes that have been annotated with labels spanning over 6000 categories. In this dataset generation, we do not use annotations. As illustrated in Fig. 3, we use several random sample color calibration charts in the market as the foreground to generate our dataset, which can generalize to arbitrary color calibration charts after learning.

With the prior knowledge that color calibration charts are rectangular, we apply random homography transformation on the foreground and guarantee it is in the proper ratio with a high probability. After that, several kinds of data augmentations like noise, exposure, and gamma are applied. Then we put the foreground onto the background with smoothing edge transitions to keep them look natural. In the last, noise, exposure, gamma, white balance, and vignetting effect are randomly applied to the combined image to simulate the extreme conditions in the wild. As illustrated in Fig. 4, we could generate infinite images as training data. With this kind



Fig. 4. Sample generated images of our dataset.

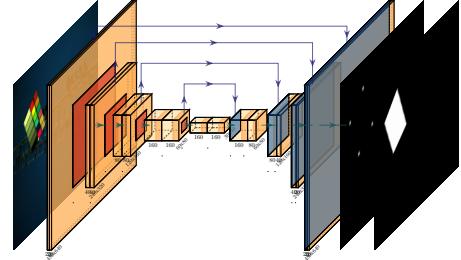


Fig. 5. UNet model structure.

of generation, our dataset provides sufficient supervision to train our neural model, which can generalize to in-the-wild scenarios.

IV. METHOD

Our scheme achieves fast and robust camera calibration only using mobile computing and RGB stream. First, a neural model is introduced and trained for color calibration chart recognition and localization (Sec. IV-A). Then, a heuristic approach based on the traditional method is adopted to automatically configure color calibration (Sec. IV-B). We also introduce a novel method to reduce the vignetting effect of the lens by human interaction (Sec. IV-C).

Notations. To achieve high performance, we adopt the UNet [6] architecture to the RGB stream sequentially obtained by the camera, as illustrated in Fig. 5. Given an input image \mathbf{I} , our network Φ predicts both a binary image $\hat{\mathbf{B}}$ and a corresponding heat map $\hat{\mathbf{P}}$:

$$\hat{\mathbf{B}}, \hat{\mathbf{P}} = \Phi(\mathbf{I}). \quad (1)$$

where $\hat{\mathbf{B}}$ is a binary mask which indicates if pixel is part of the color calibration chart and $\hat{\mathbf{P}}$ is the gaussian heat map of its four corners. \mathbf{B} and \mathbf{P} will be properly defined in later sections.

A. Neural recognition and localization

The intuitive idea is to regress the probability of every pixel being part of a color calibration chart, which could be formulated as a parsing task. During training we optimize the following loss for probability regressing, which minimizes

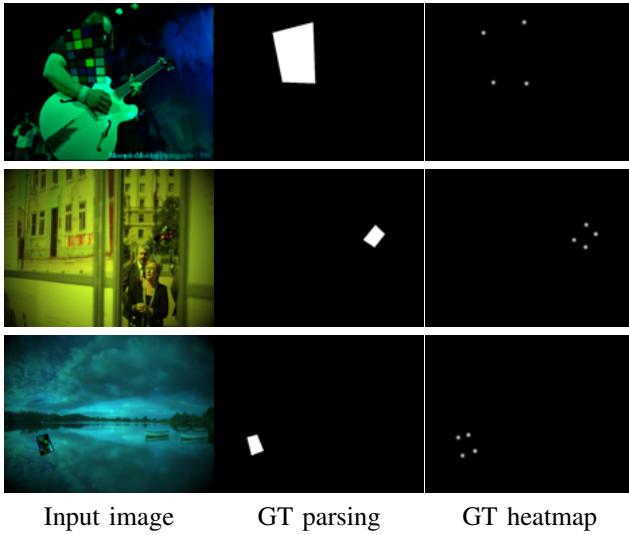


Fig. 6. Sample training data with input images and corresponding ground truth output.

the probability distance between the prediction and the ground truth from our dataset:

$$\mathcal{L}_{basic} = \|\hat{\mathbf{B}} - \mathbf{B}\|_2^2, \quad (2)$$

where \mathbf{B} is the ground truth binary parsing and $\hat{\mathbf{B}}$ is the prediction of the network.

However, only using this basic scheme fails to provide a simple way to recover the accurate coordinates of the color calibration chart. The prediction from the network will be approximately quadrilateral with a high probability, but it is still hard to recover the transformation from only a binary image, especially with much noise often. Inspired by the heatmap regression method [7], [8], which is widely used for semantic landmarks localization, we utilize a similar approach to predict the four corners of the color calibration chart. First, we construct the ground truth heatmap by putting 2D Gaussian kernels on the four corners of the generated input image, where the pixel values on the heatmaps represent the probabilities of the corresponding pixels being one of the corners. Then, we minimize the heatmap loss between the prediction and the generated ground truth:

$$\mathcal{L}_{heatmap} = \|\hat{\mathbf{P}} - \mathbf{P}\|_2^2, \quad (3)$$

where \mathbf{P} is the generated ground truth heatmap and $\hat{\mathbf{P}}$ is the prediction of network.

The final training data is illustrated in Fig. 6. During training our total loss is formulated as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{basic} + \lambda_2 \mathcal{L}_{heatmap} \quad (4)$$

where the weights for each term λ is set to be 1.0 in our experiments. Our network's parameters are optimized by Adam algorithm with an initial learning rate of 0.001. After the training process, the network is able to predict the area of the color calibration chart and output the heatmap of its corners, which could be further processed to recover the precise coordinates and transformation.

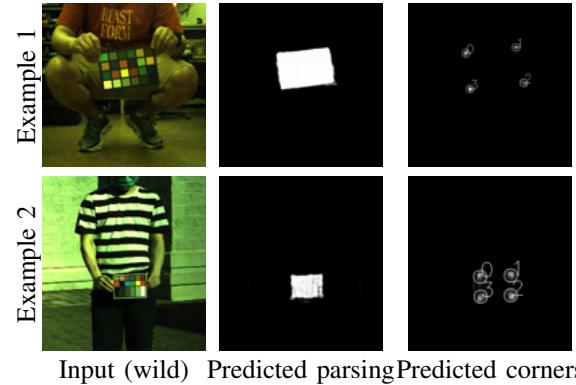


Fig. 7. Example output from network and the retrieved corners. The input image is taken in the wild.

B. Heuristic color calibration

After the network predicting the binary parsing and the heatmap, we could recover the coordinates and transformation by adopting classical image processing techniques. First, we combine both binary parsing and the heatmap to retrieve the four Gaussian kernels which have the highest possibilities to be the corners, in case sometimes the predicted heatmap has more than four gaussian kernels due to noise. Then, for each kernel we retrieved, we make a cross-correlation between it and a coordinates map, which represents the coordinates of each pixel, to obtain the precise coordinates of it. The calculation on a 640×480 image could be formulated as:

$$x = \frac{\sum \begin{matrix} \text{[Gaussian Kernel]} \\ \text{[Coordinates Map]} \end{matrix}}{\star \begin{bmatrix} 0 & 1 & 2 & \dots & 639 \\ 0 & 1 & 2 & \dots & 639 \\ 0 & 1 & 2 & \dots & 639 \\ \vdots & & & & \\ 0 & 1 & 2 & \dots & 639 \end{bmatrix}} \quad (5)$$

$$y = \frac{\sum \begin{matrix} \text{[Gaussian Kernel]} \\ \text{[Coordinates Map]} \end{matrix}}{\star \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 1 & 1 & 1 & \dots & 1 \\ 2 & 2 & 2 & \dots & 2 \\ \vdots & & & & \\ 479 & 479 & 479 & \dots & 479 \end{bmatrix}}$$

where \star represents cross-correlation. As illustrated in 7, once we get the predicted output from the network, we could precisely get the coordinates of every corner.

Once the coordinates of four corners are obtained, with the prior knowledge that color calibration charts are planar, a homography transformation \mathbf{H} could be solved to warp the color calibration chart in the input image back into normal. With the solved \mathbf{H} , we further project the input image into a nominal square, where we do not need to know the actual orientation in the input image. As illustrated in the first column in Fig. 8, the chart is approximately aligned with two axes after homography using \mathbf{H} .

To retrieve the position of color patches, a traditional method Canny edge detection [9] is applied. After the edges are obtained, we could enumerate connected components in the result to find every color chart, with the prior knowledge

that color patches are rectangular as well. Color patches often have clear edges in most cases. However, since the input images are often over blurred or underexposure, some patches could not be separated from the input images, as illustrated in the second column in Fig. 8. To retrieve the rest color patches, we would use another prior knowledge that most color patches are arranged in a grid. Then we could collect the coordinates of known color patches and apply a simple clustering method to obtain the grid interval along both the x-axis and y-axis. After that, we could reconstruct the grid and find every color patch, as illustrated in the third column in Fig. 8. Note to avoid false detection caused by wrong transformation or uncommon color patches arrangement, the predicted color patches with high RGB variance should be filtered out.

The final target is to achieve color calibration without knowing the the patches pattern of the color calibration chart. However, whatever the chart is, it must have a line of patches that form a uniform gray lightness scale. A uniform gray lightness scale means stable and strict positive or negative color gradient, which other lines of color patches could not possess. To find this specific line, we enumerate every lines in the grid and compute the color gradients of them. After filtering out the non-strict positive or negative gradients, we select the line with its gradient having the least standard deviation. The result is illustrated in the third column in Fig. 8 with bold circles.

The first and last patches in the selected line tend to be underexposed or overexposed. After filtering out the underexposed and overexposed gray patches, the value of other gray patches is obtained as $(R_i, G_i, B_i), i = 1, \dots, n$. Then the coefficients \mathbf{c} of every channel could be obtained based on the fact that gray patches should have same value on each channel in the image due to the manufacturers assure they have same value on each channel in the sRGB [10] colorspace, which is formulated as:

$$\begin{aligned} \mathbf{v} &= \begin{bmatrix} 1 \\ \frac{1}{n} \sum_{i=1}^n R_i \\ \frac{1}{n} \sum_{i=1}^n G_i \\ \frac{1}{n} \sum_{i=1}^n B_i \end{bmatrix}, \\ \mathbf{c} &= \frac{\mathbf{v}}{\|\mathbf{v}\|_{-\infty}}, \end{aligned} \quad (6)$$

where $\|\cdot\|_{-\infty}$ represents the minimum numerical value. As illustrated in the right most column in Fig. 8, a simple corrected image $\bar{\mathbf{I}}$ could be obtained by:

$$\bar{\mathbf{I}} = \mathbf{c} \otimes \mathbf{I}, \quad (7)$$

where \mathbf{c} is the per channel coefficients and \otimes represents per channel multiplication.

C. Vignetting effect reduction

Vignetting effect is a reduction of brightness toward the periphery compared to the center of the input image, which

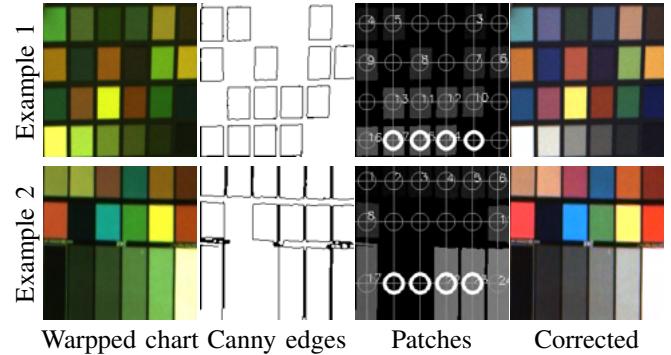


Fig. 8. Example color patches detection. The input image is taken in the wild.

could be formulated as:

$$\mathbf{I} = \mathbf{V} \odot \mathbf{I}^*, \quad (8)$$

where \odot represents element-wise multiplication, \mathbf{I} is the image with vignetting effect, \mathbf{I}^* is the ideal image without vignetting effect, and \mathbf{V} is the true vignetting field. Without the loss of generality, we assume \mathbf{V} has only one single channel. To approximate the true vignetting field \mathbf{V} , we construct $\hat{\mathbf{V}}$ by a six-even-order polynomial:

$$(\hat{\mathbf{V}})_{ij} = \alpha_0 + \alpha_1 d_{ij}^2 + \alpha_2 d_{ij}^4 + \alpha_3 d_{ij}^6, \quad (9)$$

where d_{ij} is the distance to image center of the pixel in the i th row and j th column.

We request the user to move the color calibration chart in front of the camera to complete this process. Whenever the color calibration chart is recognized and localized, a pair of (x_i, y_i) is recorded, where x_i is the distance between the color calibration chart and the image center and y_i is the average brightness of the whole chart. Through the recorded data, least square method is applied to regress α , which could be formulated as:

$$\begin{aligned} \min_{\alpha} \quad & \|\mathbf{X}\alpha - \mathbf{y}\|_2^2 + \lambda \alpha^T \alpha, \\ \text{s.t.} \quad & \mathbf{X} = \begin{bmatrix} 1 & x_1^2 & x_1^4 & x_1^6 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_n^2 & x_n^4 & x_n^6 \end{bmatrix}, \\ & \mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}. \end{aligned} \quad (10)$$

Since least square solver often takes a time complexity of $O(kn)$ or $O(n^2)$, which would be slow when the total data is getting larger and larger if we try to preserve all the data point, we prefer a method that has constant cost each time in practice. The incremental least square method could be formulated as:

$$\begin{aligned} \mathbf{A}_0 &= \lambda \mathbf{E}_{4 \times 4}, \\ \mathbf{b}_0 &= \mathbf{0}_{4 \times 1}, \end{aligned} \quad (11)$$

$$\begin{aligned} \mathbf{A}_k &= \mathbf{A}_{k-1} + \mathbf{X}_k^T \mathbf{X}_k, \\ \mathbf{b}_k &= \mathbf{b}_{k-1} + \mathbf{X}_k^T \mathbf{y}_k, \\ \boldsymbol{\alpha}_k &= \mathbf{A}_k^\dagger \mathbf{b}_k, \end{aligned} \quad (12)$$

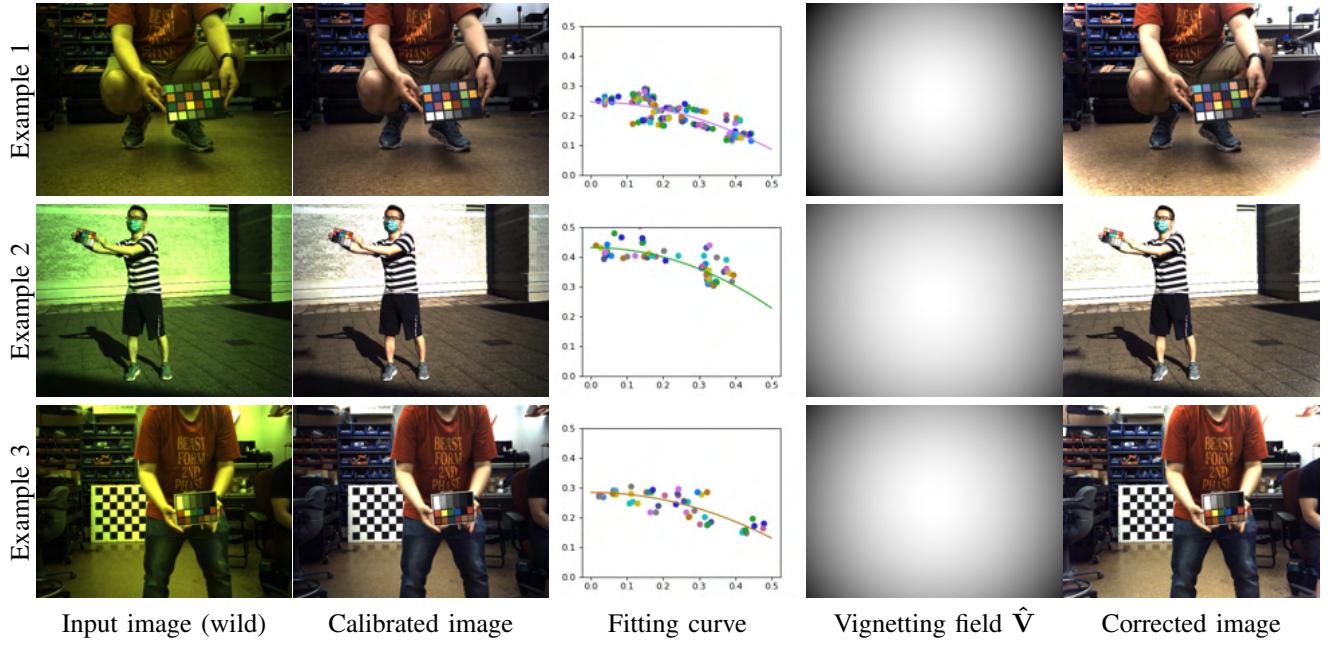


Fig. 9. Example of reducing vignetting effect. The input images are taken in the wild.

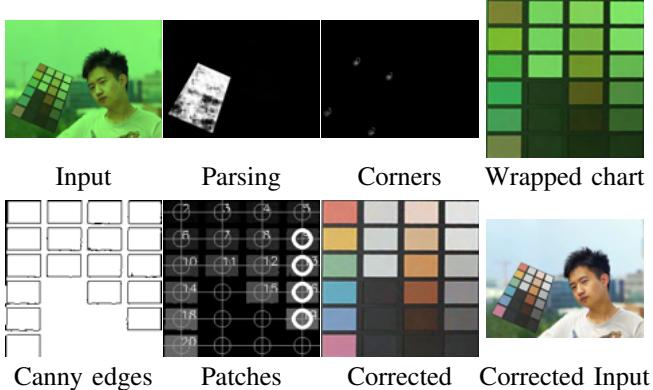


Fig. 10. Example of applying our method on arbitrary color chart which is never seen before.

where \dagger represents the pseudo inverse of matrix, which takes a time complexity of $O(1)$ each time to update. As shown in Fig. 9, the corrected image significantly reduce the vignetting effect.

V. EXPERIMENT

Here we evaluate our method in various challenging scenarios. We run our experiments on an edge device with Intel NUC, where our model achieves recognition and localization on 4112×3008 RGB input in 160 ms, and further processing finishes in 100 ms. Fig. 11 demonstrates several more results of our approach, which can successfully correct the color in the wild. As shown in Fig. 11, blurring, overexposure and extreme distance could significantly reduce the rate for successful calibration.

As illustrated in Fig. 10, our method could directly apply on a new color calibration chart which is not in the dataset, as long as it has the similar pattern like other charts.

Note that under some scenes, the result after color calibration and vignetting reduction may seem overexposed. It could be avoided if we multiply a constant on all the channels in practical use.

VI. CONCLUSION

Limitations. As a trial to explore neural and heuristic methods, our approach still owns limitations as follows. First, our approach cannot handle the input with extreme motion blur, like images from a long exposure time, where patches are hard to retrieve. Besides, the model fails to recognize charts when they are at extreme distances. The performance of computing time of our approach on edge devices could be improved in the future.

Conclusion. We have presented a fast neural color calibration chart recognition and localization approach, a heuristic approach to enable automatic color calibration, and a novel approach to reduce the vignetting effect. Our method could work in the wild without prior knowledge of the appearance of the color calibration card. We believe that our approach will make the color calibration process much easier.

ACKNOWLEDGEMENT

The authors would like to thank the Robotics Institute Summer Scholars Program for providing the opportunity to work in the AirLab as well as my mentors Yaoyu Hu and Sebastian Scherer for their guidance throughout this summer. We also thank Rachel Burcin and John Dolan for their effort to bring this whole program online.

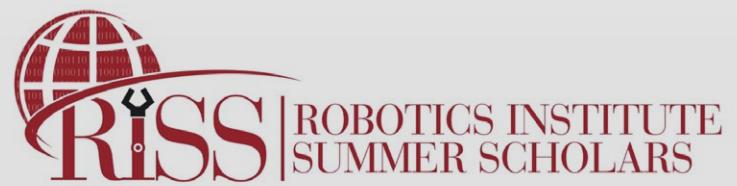
REFERENCES

- [1] P. Bergmann, R. Wang, and D. Cremers, “Online photometric calibration of auto exposure video for realtime visual odometry and slam,” *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 627–634, 2017.

Description	Input image (wild)	Calibrated image	Vignetting	Corrected image	Frames calibrated
indoor blurred					384/747
outdoor far					292/626
outdoor white base					179/207
outdoor overexposed rotating					199/375
outdoor near overexposed					240/309
outdoor rotating					245/264
outdoor near rotating					140/227

Fig. 11. More results on difference scenes. ‘Frames calibrated’ represents a ratio from the number of frames that could be successfully recognized and calibrated to the total number of all frames.

- [2] K. Barnard, “Practical color constancy,” PhD thesis, Simon Fraser University, School of Computing, 1999.
- [3] Y.-C. Liu, W.-H. Chan, and Y.-Q. Chen, “Automatic white balance for digital still camera,” *IEEE Transactions on Consumer Electronics*, vol. 41, no. 3, pp. 460–466, 1995.
- [4] A. Kordecki, H. Palus, and A. Bal, “Fast vignetting reduction method for digital still camera,” in *2015 20th International Conference on Methods and Models in Automation and Robotics (MMAR)*, 2015, pp. 1145–1150.
- [5] cvdfoundation, “Open images challenge 2018/2019 test set,” <https://github.com/cvdfoundation/open-images-dataset#download-the-open-images-challenge-20182019-test-set>.
- [6] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.
- [7] T. Pfister, J. Charles, and A. Zisserman, “Flowing convnets for human pose estimation in videos,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1913–1921.
- [8] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler, “Joint training of a convolutional network and a graphical model for human pose estimation,” *Advances in neural information processing systems*, vol. 27, pp. 1799–1807, 2014.
- [9] J. Canny, “A computational approach to edge detection,” *IEEE Transactions on pattern analysis and machine intelligence*, no. 6, pp. 679–698, 1986.
- [10] Wikipedia, “srgb,” <https://en.wikipedia.org/wiki/SRGB>.
- [11] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, “Openpose: Realtime multi-person 2d pose estimation using part affinity fields,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, pp. 172–186, 2021.
- [12] R. P.I and D. V.S, “License plate localization: A review,” *international journal of engineering trends and technology*, vol. 10, pp. 604–615, 2014.
- [13] Z. Luo, Z. Wang, Y. Huang, T. Tan, and E. Zhou, “Rethinking the heatmap regression for bottom-up human pose estimation,” 2020.



riss.ri.cmu.edu

Carnegie Mellon University
The Robotics Institute

Carnegie Mellon University
School of Computer Science